Tangible objects para mejorar habilidades espaciales en entornos 3D
Tangible objects to improve spatial abilities in 3D environments

Trabajo de Fin de Máster
Curso 2019-2020

Autor
Michael Tomé Rodríguez

Director
Jorge Jesús Gómez Sanz

Máster en Ingeniería Informática
Facultad de Informática
Universidad Complutense de Madrid

Convocatoria: Junio 2020
Calificación: 9

Tangible objects para mejorar habilidades espaciales en entornos 3D
Tangible objects to improve spatial abilities in 3D environments

Trabajo de Fin de Máster en Ingeniería Informática
Departamento de Ingeniería del Software e Inteligencia Artificial

Autor
Michael Tomé Rodríguez

Director
Jorge Jesús Gómez Sanz

Convocatoria: Junio 2020
Calificación: 9

Máster en Ingeniería Informática
Facultad de Informática
Universidad Complutense de Madrid

Agradecimientos

Quisiera agradecer a varias personas y entidades la ayuda y el apoyo que me han prestado para la realización de este Trabajo de Fin de Máster.

En primer lugar, a mi director Jorge J. Gómez Sanz por confiar en mí para la realización de esta tesis y por su inestimable ayuda. Ante todas las dificultades que se han presentado, estando dispuesto a dedicarme su tiempo, incluyendo su tiempo libre, y su conocimiento.

También agradecer al proyecto 'Diseño colaborativo para la promoción del bienestar en ciudades inteligentes inclusivas (TIN2017-88327-R)' por cederme el hardware necesario sin el que éste Trabajo de Fin de Máster no hubiese sido posible.

Agradecer también a mi familia, concretamente a mi madre y a mi hermano, sin ellos no podría haber seguido estudiando ni la carrera ni el máster. Su apoyo emocional me ha permitido seguir adelante en los momentos más duros.

Por último, pero no menos importante, agradecer a mi abuelo materno, que en paz descanse, por haber dedicado las últimas palabras que me dijo a apoyarme y decirme que siguiera esforzándome por ser alguien del que yo mismo sentirme orgulloso.

A todos, de todo corazón, os lo agradezco.

Resumen en castellano

Muchas personas tienen problemas a la hora de interactuar con entornos 3D por dificultades en la percepción de la profundidad. Este trabajo propone el diseño y desarrollo de un framework de objetos tangibles (tangible objects). En el trabajo se construye un prototipo que permite interactuar con un entorno 3D usando como referencia el mundo real. Combina tecnologías del Internet de las cosas junto a servidores basados en angular. Con estos elementos, se logra construir un sistema de bajo coste. La ventaja de este objeto tangible es que puede tener cualquier forma y ser ajustable a la forma de interacción con el entorno. El entorno se pone a prueba en un caso de estudio donde se compara el movimiento de los elementos en una escena 3D con movimientos en el mundo físico.

Palabras clave

Objetos tangibles, entorno 3D, pantalla, ordenador, habilidades espaciales.

Abstract

Many people have problems interacting with 3D environments due to difficulties on depth perception. This work prpose the design and development of a framework of tangible objects. In this work it's built a prototype that allows to interact with 3D evironments using the real world as reference. It combines technologies of Internet of things with angular based servers. With this elements is possible to create a low cost system. The advantage of this tangible object is that it can have any shape and be adjustable to the way of interaction with the environment. The system is tested in a case study where the movement of the elements in a 3D scene is compared to physical world movements.

Keywords

Tangible objects, 3D environment, screen, computer, spatial abilities.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción a los tangible objects

1.1. Introducción

Este proyecto está motivado por las dificultades que pueden tener algunas personas para comprender cómo pueden existir objetos con formas específicas en un entorno que a priori es plano, como lo es una pantalla de ordenador y cómo puede existir un espacio tridimensional en las mismas. Este tipo de problemas se puede dar tanto en niños que no hayan tenido interacción con un ordenador como en personas mayores que comiencen el uso del mismo, así como de personas que tienen un problema de logopedia o que tienen ciertos problemas de visión. Estos problemas de visión son denominados como problemas de visión estereoscópica, disparidad binocular o una deficiencia en la percepción de la profundidad, lo que significan la falta de estereopsis[21][53], es decir, una deficiencia del desarrollo de la capacidad no innata de formar imágenes tridimensionales a partir de los ojos, lo que se traduce en la falta de capacidad de percibir de forma correcta la profundidad y la percepción en tres dimensiones en condiciones desfavorables. Esto puede implicar dificultades cuando la persona intenta conducir con seguridad, realizar trabajos o ejercicios que requieran una buena precisión o disfrutar y trabajar en la industria de los videojuegos.

En el día a día, aún sin tener esta capacidad, el cerebro desarrolla ciertos métodos haciendo uso de la iluminación, las sombras, objetos de referencia, el volumen y el tamaño para suplir este problema. También se da el caso de inclinar la cabeza para enfocar y percibir

correctamente ciertos objetos. Sin embargo, cuando se trata de aplicar estos métodos a entornos 3D, no funciona de la forma que debería porque estos entornos no tienen por qué tener sombras, iluminación, las distancias y tamaños podrían no corresponderse con los que se está acostumbrado en el mundo real y la profundidad misma puede ser engañosa.

Para resolver este problema se propone aprovechar el concepto de Tangible Object u Objeto Tangible[12]. Este tipo de elemento podría proveer de una forma más intuitiva de traducir lo que ocurre en el mundo real a una escena 3D y viceversa.

1.2. Objetivos

La realización de este proyecto tiene como principal objetivo la creación y utilización de un tangible object u objeto tangible que, como su nombre indica, es un objeto que pueden ser tocado y manipulado físicamente y que permiten interactuar de una u otra forma con un elemento digital. Así, con un objeto tangible se puede representar un objeto tridimensional dentro de una pantalla de ordenador con el fin de mejorar o desarrollar un mejor entendimiento de los entornos tridimensionales. También para ayudar en la mejora de las habilidades espaciales y el reconocimiento de los objetos en estos entornos.

Con estos objetos tangibles se intenta simular la realidad dentro del ordenador y comprobar la capacidad de profundidad y de geometría e intentar mejorar estas capacidades haciendo una relación entre objeto y modelo en la pantalla del ordenador.

Con la relación entre objeto tangible y modelo en la pantalla, el objetivo principal es buscar ayuda al manejar la profundidad, las distancias y la capacidad de reconocer las formas reales en los entornos tridimensionales mediante la similitud de estos objetos tangibles y sus modelos y la imitación de forma y movimiento de los últimos. Con la posibilidad de los objetos tangibles de cambiar su forma, se puede hacer que la inmersión sea mayor y permita un mejor entendimiento de la profundidad.

Para conseguir este objetivo, el objeto tangible debería de ser capaz de ofrecer la información de sus movimientos y enviarlos al ordenador al que esté conectado.

Como objetivo secundario, se busca crear un sistema que permita que el desarrollo de soluciones que hagan uso de un objeto físico y su modelo virtual sea más sencillo. Para lo cual el sistema necesita de un framework modificable y reutilizable. Además de una estructura adecuada.

En resumen, los objetivos definidos son:

- Crear un objeto tangible de bajo coste.
- Hacer que el objeto tangible y su representación en pantalla tengan la misma forma y se pueda cambiar.
- Identificar los gestos y movimientos del objeto tangible para que su representación en pantalla los imite y crear una relación de inmersión.
- Crear un framework para acelerar el desarrollo de aplicaciones que usen tangibles.

Con los objetivos definidos, el proyecto trata de relacionar los objetos físicos con su modelado en la pantalla para ayudar a mejorar. Con esa idea en mente, el sistema recibe el nombre de **HITO** (*Help to Improve with Tangible Objects*).

1.3. Plan de trabajo

Esta memoria detalla el desarrollo de un sistema con tangible object, un objeto físico que se traslada a un entorno tridimensional dentro de la pantalla de un ordenador, permitiendo reflejar los movimientos del tangible object en su reflejo digital.

El desarrollo del proyecto se ha definido en dos partes, primero una definición del producto y luego una explicación del mismo definiéndolo en sus puntos más básicos. Para ello,

se han definido las siguientes fases:

1. **Estado del arte:** En esta fase se da a conocer otros trabajos relacionados con los tangible objects. Cubrirá un análisis de la competencia dividido en los proyectos que usan objetos para las interacciones y las plataformas que hacen uso de los mismos. Además se analizarán las tecnologías que se han descartado y se han escogido.

2. **Requisitos del sistema:** Esta fase comprende toda la investigación de los usuarios y el diseño del proyecto. Los usuarios son inventados para reflejar el público objetivo de este producto.

    2.1. **Investigación del usuario:** Se buscan los posibles usuarios a partir de la premisa del proyecto, personas con problemas en el entendimiento de los entornos tridimensionales.

    2.2. **Modelado:** A partir de los datos anteriores, se crean personas ficticias que representan los usuarios objetivo y se plantean situaciones en los que el sistema podría ser de utilidad, obteniendo los objetivos del sistema.

    2.3. **Requisitos:** A partir de los usuarios objetivos encontrados y los objetivos descubiertos, se definen los requisitos que ha de cumplir el sistema.

    2.4. **Descripción del producto:** Con los conocimientos anteriores, se crean un concepto general de HITO. Se define el framework de interacción que permite crear un prototipo visual y físico del producto. Se realiza mediante iteraciones para incrementar la fidelidad de los prototipos.

    2.5. **Casos de uso:** Una vez definido el framework, se definen las posibilidades que ofrece el sistema para interactuar con el objeto tangible.

Una vez definido el producto, se procederá a construirlo y a probarlo:

1. **Arquitectura del sistema:** Teniendo todos los datos anteriores, se procede al desarrollo del sistema, describiendo la arquitectura y sus componentes.

2. **Implementación del sistema:** Después de la arquitectura, se procede a explicar la implementación utilizada.

3. **Experimentación:** Se plantean descripciones detalladas de cómo se sentiría utilizar el sistema en cada momento.

1.4. Estructura del documento

Para estructurar el plan de trabajo del apartado anterior, se plantea una estructura para el documento que reúna los puntos más importantes como capítulos y las partes de estos capítulos como secciones de los mismos. Además, se añaden anexos con información relevante para el proyecto como documentos adicionales que no pertenecen al desarrollo de los capítulos del plan de trabajo.

Este mismo capítulo trata de dar la información necesaria para comprender los objetivos y motivos de la realización del proyecto.

En el capítulo 2 se estudia, como se ha dicho en la sección 1.3, el avance tecnológico con respecto al proyecto. Dividiendo estos en proyectos con tangible objects en la sección 2.1 y de plataformas que tengan estos objetos en la sección 2.2. Además de mostrar las tecnologías que se ha rechazado al plantear el proyecto en la sección 2.4.

En el capítulo 3 se definen los requisitos del sistema realizando una investigación de los usuarios en la sección 3.1 junto al modelado en la sección 3.2, de personas ficticias, el establecimiento de los requisitos en la sección 3.3 y la definición del producto en la sección 3.4. Además de los casos de uso del sistema en la sección 3.4.5. También se establecen las tecnologías elegidas en la sección 2.5.

En capítulo 4, a partir de los resultados del estado del arte y los requisitos del sistema, se comienza el desarrollo del sistema, donde se explica la arquitectura en la sección 4.1 y el

diseño del mismo en la sección 4.2.

Separado del anterior capítulo, está el capítulo 5, la implementación en la cual se desarrolla y explican los elementos que componen el sistema completo, separando lo desarrollado en el ordenador, en la sección 5.1, del objeto tangible en la sección 5.2.

Como último capítulo, el 6 del plan de trabajo, se expone una experimentación con ejemplos de uso del sistema desarrollado.

En el capítulo 7, último de la memoria, se detallan las conclusiones obtenidas al finalizar el desarrollo del proyecto y se plantea el trabajo futuro para mejorar o añadir nuevos elementos al proyecto.

Las traducciones de la introducción y de las conclusiones se encuentran en los anexos A y B respectivamente. También se encuentran los manuales para el despliegue del sistema, en el anexo C, para saber cómo hacer funcionar el mismo, y el de usuario para saber como utilizarlo tanto usuarios objetivos como secundarios en el anexo D.

Capítulo 2

Estado del arte

En el estado del arte se va a analizar qué proyectos terminados o en desarrollo, artículos o plataformas existentes están relacionados con los objetivos del proyecto. De los análisis se extraerán los puntos importantes para hacerlo funcionar y qué utilidad tienen con respecto al usuario objetivo.

Estos análisis de la competencia con respecto al proyecto, se van a dividir en dos partes: aquellas tecnologías que tengan que ver con el desarrollo de tangible objects; aquellas plataformas que hagan uso de los tangible objects.

Como punto final, después de los análisis, se recogerán las tecnologías investigadas y descartadas que podían ayudar con el desarrollo, buscando elementos que hagan de objetos tangibles y que puedan dar información de sus movimientos a través de sensores.

Los puntos anteriores resultarán en un acercamiento a los objetivos que se buscan.

2.1. Tangible Objects

Hay una mucha variedad de proyectos o artículos que utilizan la idea de un objeto tangible. A continuación, se listan varios:

- **Digital Desktop**[9]: se ha pensado en crear un escritorio digital controlado por un

proyector y una cámara que lee los movimientos de las manos para reaccionar a qué objetos proyectados y digitales intentan tocar. Con este sistema se logra imitar todo aquello que exista en un escritorio normal y corriente. Las manos y sus gestos son los que actuarían directamente como los objetos tangibles puesto que son éstas las que permiten que los objetos digitalizados del escritorio digital reaccionen.

Digital Desktop interacciona con las manos, mientras que HITO quiere hacerlo con un objeto físico. Se deben de poder leer los movimientos de este objeto con algún elemento, como un sensor.

- **Tangible Tele-Meeting** [54]: trata de proyectar la imagen de una persona que está siendo grabada por varias cámaras en un robot, situado donde se encuentra la otra persona de la videoconferencia, que simule los movimientos de la persona grabada así como que tenga una proyección de su aspecto. Se intenta eliminar la separación física mediante la creación de un sistema que simule a la otra persona.

  Crear el entorno resulta excesivamente caro en Teangible Tele-Meeting. En relación con el proyecto, de manera intencional, busca reconocer objetos tridimensionales o personas a través de la pantalla para que se comporten como si fueran físicos con un bajo coste.

- **Tangible Virtuality** [50]: busca la virtualización tangible que, mediante un sistema de cámaras, proyectores y unos sensores que detecten los movimientos de las manos, permitir manipular una representación en tres dimensiones de un elemento físico. Esta proyección tridimensional se vería afectada dependiendo de los gestos de las manos que tengan los sensores.

  Este artículo da un paso más allá en el concepto del escritorio digital anteriormente mencionado y de la propia idea de este proyecto. En lugar manipular un objeto digital

mediante uno físico, manipulas la representación holográfica del mismo directamente. Pero tiene el problema de que no se siente el objeto físico directamente. Con HITO, se puede tener un objeto de casi cualquier forma para mejorar la relación entre el objeto representado y el físico.

- **Tangible and Graphical**[5]: busca hacer coincidir el objeto físico con la representación virtual, construyéndolo desde cero y pieza por pieza. La interfaz planteada es simplemente ver el objeto en construcción.

  En lugar de tener el tangible object y la representación digital del mismo predefinidas, HITO trata de construir ambos al mismo tiempo. Al modificar el objeto físico, su representación digital cambia.

- **Digital Twins**[14]: los gemelos digitales buscan imitar una escena completa real en una escena 3D, duplicando objetos y simulando lo que ocurre en la escena real.

  En lugar de imitar por completo la escena, en HITO lo que se busca es interaccionar con la escena para aumentar la inmersión y la percepción de la profundidad.

Como se puede observar, algunos de los artículos científicos expuestos tienen como objetivo hacer una interacción más real entre un tangible object y un elemento digital. Sin embargo, se asume que las personas entienden las representaciones digitales de objetos reales sin necesidad de un aprendizaje inicial.

Para cumplir los objetivos de HITO, se ve la necesidad de un objeto físico que pueda ser manipulado y del que además se puedan leer sus movimientos de alguna forma. Como en el proyecto no se plantea el uso de una pantalla táctil o similar debido a que podría aumentar el coste, el propio objeto físico debería poder dar la información de sus movimientos. Para leer la información es necesario que este elemento contenga sensores y que puedan ser leídos por el ordenador. Para contener estos sensores en el objeto tangible, el objeto físico ha de ser el

que pueda leer su información y enviarlas. Un elemento que cumpla estas condiciones es un microordenador[52] o un ordenador en miniatura, ya que es manejable debido a su tamaño y que tiene la capacidad de procesamiento suficiente como para procesar la información de los sensores que se le agreguen sin problemas.

2.2. Plataformas tecnológicas de tangible objects

Se han revisado trabajos de investigación generales sobre tangible objects. En esta sección se revisa el problema de crear de forma disciplinada este tipo de objetos usando plataformas.

Las plataformas creadas en torno a los objetos tangibles son variadas y hacen un uso distinto de los objetos tangibles para cumplir con su función. Estas plataformas pueden ayudar a formar una idea de cómo abordar el uso de estos objetos tangibles en el proyecto.

2.2.1. Osmo

Figura 2.1: Osmo: Learning games

Osmo[34] es un aplicación educativa que contiene diferentes tipos de minijuegos que ayudan a los niños a aprender de forma divertida y original. Osmo está desarrollado para iPad y necesita de un escáner y varios elementos para los distintos tipos de juegos. El escáner, una de las piezas más importantes del sistema, se encarga de integrar los objetos que ve para aplicarlos al juego que esté en ejecución en ese momento, como por ejemplo, en el juego de puzzles, éste se encarga de reconocer si la posición de las piezas coinciden con la imagen mostrada a representar. En otro ejemplo, el escáner puede hacer que los objetos que dibujes en un papel tengan su representación en la pantalla en tiempo real.

- **Ventajas:** Es una herramienta divertida y entretenida que además permite enseñar a los niños de forma sencilla tanto matemáticas, como puzzles, codificación simple, etc además de permitir relacionar objetos físicos reales con elementos digitales, como es en el caso de los puzzles. La interfaz depende de los juegos, siempre es sencilla.
- **Desventajas:** Es exclusivo para iPad, lo que reduce el número de personas que pueden acceder a ello. También se tiene su precio, de 119,95 euros, bastante elevado, pero teniendo en cuenta que es un kit completo para la aplicación.

2.2.2. Quiver

Figura 2.2: App de Quiver

Quiver[38] es una aplicación gratuita diseñada específicamente para el uso de la realidad aumentada. Con esta aplicación se puede dar vida a dibujos de sus plantillas, teniendo en físico el papel con el dibujo pintado y viendo a través de la pantalla del smartphone ese mismo dibujo pero en tres dimensiones; incluyendo ciertas animaciones. Los dibujos pueden ser tanto sencillos como complejos, al igual que las animaciones mostradas por la aplicación. Tiene un coste de 4,79 euros en la versión de Android.

- **Ventajas:** Permite de forma sencilla observar dibujos planos en forma tridimensional de forma sencilla. Es amigable con los niños y permite aprender de forma divertida. Las formas son complejas y poseen animaciones. La interfaz es muy sencilla, contando con unos pocos botones y siendo el centro de la pantalla lo que ve la cámara.

- **Desventajas:** Su objetivo es el paso de un dibujo físico plano a un objeto digital en forma tridimensional. difiriendo de la intención del proyecto de representar un objeto tridimensional en otro en formato digital. Está limitada a las plantillas que la empresa tiene creadas.

2.2.3. TOR – Tangible Object Recognition

Figura 2.3: TOR

TOR[47] es una herramienta que permite, como su nombre indica, el reconocimiento de objetos físico y mostrarlos en una pantalla.[1] Lo que se muestra en dicha pantalla es la base del mismo objeto, ya que no posee un escáner de tres dimensiones. La herramienta en sí es una pantalla grande que reconoce la base del objeto apoyado en la misma.

- **Ventajas:** Es una herramienta que reconoce los objetos que se ponen en su pantalla y los representa como su base en la pantalla del ordenador. La interfaz es sencilla y reacciona cuando un objeto es colocado en la pantalla.
- **Desventajas:** Es una herramienta muy grande y de alto coste. Su función básica es muy reducida y necesitas tanto la herramienta como un ordenador para poder utilizarlo, sin poder usar la propia pantalla de TOR para otra cosa que no sea el reconocimiento de las bases de los objetos.

2.2.4. Playstation Move

Figura 2.4: PS Move

PS Move[35] es un sistema de control de movimiento pensado para la Playstation 3 y 4. Este sistema está dirigido hacia la realidad virtual y se complementa con las gafas de realidad virtual. El sistema permite tanto realizar acciones en los juegos como imitar un objeto de los mismos y controlarlos. El ejemplo más claro es tomar la forma de un arma en un shooter. Además posee adaptadores que hacen que imite más aún un arma de los juegos.

- **Ventajas:** Permite mover y rotar un objeto del que imita la forma.
- **Desventajas:** Es un sistema de alto coste y dirigido solo a dos consolas específicas.

2.2.5. Nintendo Labo

Figura 2.5: Nintendo Labo

Nintendo Labo[32] es una plataforma de juguetes y juegos de construcción desarrollada por Nintendo para Nintendo Switch. Se trata de una serie de juegos orientados a los niños en los que interactúan con elementos físicos para generar reacciones en los juegos de la Switch. Estos elementos físicos son construcciones de cartón diseñadas para desempeñar la acción que se quiere simular, como es el caso de una caña de pescar, colocando un relé de cartón para aparentar estar pescando.

- **Ventajas:**

  Estimula la relación entre objeto físico y virtual en los niños, ayudando a la coordinación ojo-mano

  Ofrece interacciones entre el objeto simulado y los objetos del juego

- **Desventajas:**

  Se necesita una Nintendo Switch y Nintendo Labo se vende por separado, lo que supone un alto coste

  Las figuras son las predefinidas el contenido de los cartones

2.2.6. Tangible Graph Builder

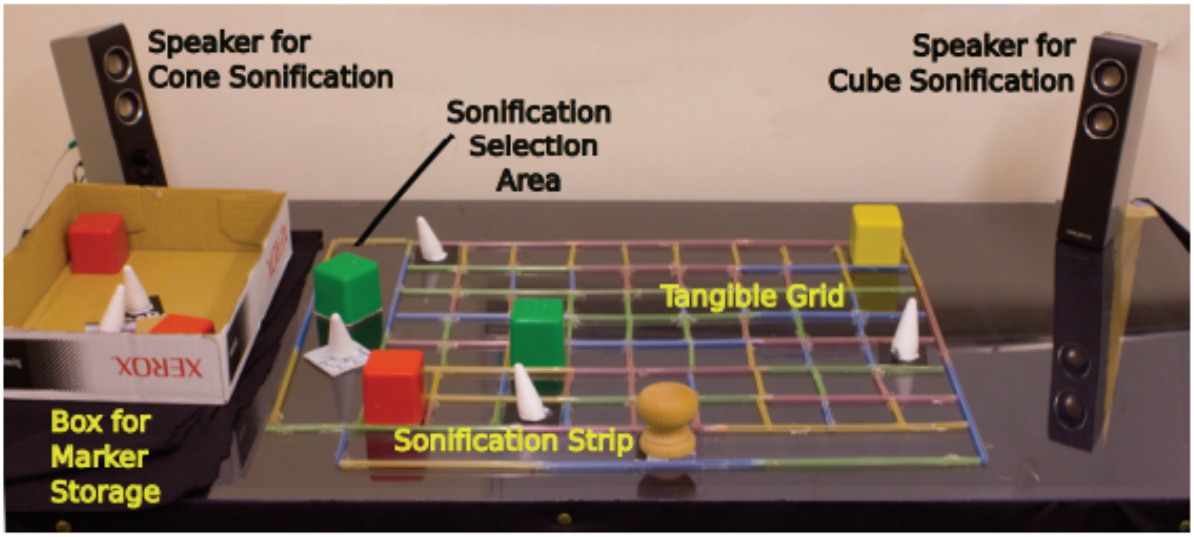


Figura 2.6: Tangible Graph Builder

Tangible Graph Builder[28] es una interfaz de usuario tangible ideada por David McGookin, Euan Robertson, Stephen Brewster de la Universidad de Glasgow, E.E.U.U., para la creación digital de gráficas mediante objetos físicos. La idea radica en las posibilidades de poder crear distintos tipos de gráficas sin necesidad de utilizar un programa específico con sus reglas y formas de hacer sus gráficas.

- **Ventajas:**

  Trata de relacionar objeto físico con un resultado en digital

  Los objetos físicos se distinguen unos de otros fácilmente

- **Desventajas:**

  En el momento de esta idea (2010) la tecnología no estaba preparada para poder llevar acabo esta idea

  La cantidad de objetos físicos es limitada para gráficas grandes

Se necesita una retroalimentación para saber que el objeto está leído y colocado, como el sonido

El espacio de trabajo para la gráfica puede ser muy grande dependiendo de la misma.

2.3. Resumen de trabajos revisados

De análisis de la competencia se pueden extraer información que ayudarán a orientar el proyecto para cumplir los objetivos. La tabla 2.1 comprende los resultados tanto de la sección 2.1 como de la sección 2.2.

Digital Desktop	Imitar el movimiento de un objeto físico en un objeto virtualizado La interfaz es la misma que la de un escritorio de ordenador normal
Tangible Tele-Meeting	Representar el objeto físico
Tangible Virtuality	Imitar el movimiento del objeto físico virtualizado
Tangible and Graphical	Cambiar la forma del objeto tanto físico como modelo representado La interfaz presenta el objeto creado
Osmo	Gran variedad de contenido Empareja objetos físicos con objetos digitales La interfaz es sencilla
Quiver	Enseña la correlación entre objeto plano y objeto digital en 3D
TOR	Emparejamiento de objeto físico a digital La interfaz cambia cuando un objeto interactúa con la pantalla
PS Move	Un modelo 3D imita los movimientos del sistema La interfaz depende del juego
Nintendo Labo	Ayuda a la coordinación ojo-mano Ofrece interacciones entre objeto modelado y objetos del juego Las interfaces son amigables y sencillas
Tangible Graph Builder	Relación entre objeto físico y resultado digital Necesidad de una retroalimentación

Cuadro 2.1: Tabla de puntos extraidos de los análisis

2.4. Alternativas tecnológicas descartadas

A continuación se enumeran y describen las tecnologías, tanto hardware como software, que han sido descartadas a lo largo del desarrollo del proyecto. Estos descartes se deben a que no se ajustaban a lo que se pensaba que hacía falta.

En cuanto a hardware se buscaba un microordenador con una alta capacidad de procesamiento para poder recibir y enviar datos en tiempo real sin bloquearse, que tuviese sensores compatibles y pudiese utilizar varios de forma simultánea. Estos sensores tendrían que ser capaces de leer los movimientos del microordenador.

2.4.1. Hardware

En esta sección se enumeran y describen los elementos físicos descartados en el desarrollo, tanto placas como sensores.

- **Raspberry Pi**[17]: Es un microordenador de tamaño reducido y bajo coste desarrollada por la fundación Raspberri Pi. En sus versiones 3 y 4 superan a ordenadores con poca potencia de hace algunos años. Estas versiones poseen conexión inalámbrica a internet, un buen número de puertos USB y una frecuencia de reloj, RAM y un procesador gráfico altos en comparación con el precio y el rendimiento otorgados. Funciona tanto con lenguajes de programación de bajo nivel como Python o C, como con los de alto nivel como Java o C++. Tiene buena cantidad de componentes, pero todos desarrollados por la misma empresa. Tiene una comunidad amplia que desarrolla proyectos constantemente. Fue descartada debido a que ya había hecho uso de alguna para algún proyecto personal.

  - **Ventajas:**

    Variedad de componentes

    Gran comunidad.

    Bajo coste.

- **Defectos:**

    Se buscaba una placa que no se haya utilizado previamente.

- **Arduino**[7]: Es un microordenador de código abierto que posee una gran cantidad de variantes, tanto de hardware como en la forma de la misma, pudiendo ser más alargada o cuadrada. La diferencia en hardware radica en la diferencia de memoria RAM, la diferencia de conexiones USB, la cantidad de pines, la potencia del procesador, los componentes de conexión a internet y más variedad.

    - **Ventajas:**

        Altamente utilizado.

        Gran comunidad.

        Bajo coste.

        Variedad de componentes

    - **Defectos:**

        Limitado en funcionalidad y es preferible no usarlo como prototipo.

- **MinnowBoard MAX**[15]: Una placa creada por Intel y CircuitCo. Tiene varias versiones, diferenciándose en su funcionalidad y también en su precio, siendo el mínimo bastante mayor al de otras alternativas.

    - **Ventajas:**

        Varias versiones

    - **Defectos:**

        Alto coste.

        Poca disponibilidad.

- **Sensor visual: cámaras:** No un sensor en sí, sino una herramienta que permitiría 'darle ojos' al proyecto para ver el objeto tangible.

- **Ventajas:**

  Funciona como un detector de movimiento, permitiendo ver los movimientos del objeto tangible.

- **Defectos:**

  Aumentaría el tamaño del objeto tangible.

  Si estuviese en un lado fijo obligaría a estar dentro de su campo de visión. datos.

  Se necesitan varias para controlar todos los movimientos en las 3 dimensiones.

  No hay compatibilidad directa entre la interfaz Grove[41].

- **Sensor: acelerómetro analógico:** un componente con interfaz Grove que mide la aceleración teniendo en cuenta la gravedad de la Tierra. Mide en 3 ejes digitales y hasta16 g. Es de bajo consumo.

  - **Ventajas:**

    Tiene mucha precisión.

    Es un elemento físico muy pequeño y no requiere de mucha capacidad de procesamiento.

  - **Defectos:**

    Tiene que funcionar con I2C, no funciona con UART.

    Del chip utilizado concretamente, tiene una interfaz de Grove, por lo que solo es compatible con placas que tengan esta interfaz.

    Es incapaz de medir la aceleración al girarlo en sobre sí mismo en horizontal.

    Si no se controla adecuadamente puede provocar bloqueos en el ordenador.

    Es demasiado sensible a los cambios.

El principal motivo de descartar este frente a su contraparte digital fue la disponibilidad y la alta sensibilidad que podría provocar muchos errores.

- **Sensor: giroscopio analógico:** es un sensor con interfaz Grove de 3 ejes de salida digital y de bajo consumo. Es de bajo consumo.

  - **Ventajas:**

    Mide en todas las direcciones, siempre que el chip se mueve, obtiene una medición.

    Tiene una altísima precisión.

    Es muy pequeño.

    Bien controlado no requiere de mucha capacidad de procesamiento.

  - **Defectos:**

    Tiene que funcionar con I2C, no funciona con UART.

    Si se compara con su contraparte digital, la sensibilidad sería demasiado alta.

    Los resultados hay que transformarlos para obtener una medida en grados por segundo.

- **Sensor: Ultrasónico:** un sensor que permite medir las distancias entre él y el objeto al que apunte mediante ultrasonidos.

  - **Ventajas:**

    Capaz de medir las distancias entre el objeto tangible del que formaría parte y el lugar de apoyo.

    Es un elemento físico muy pequeño.

    No requiere de mucha capacidad de procesamiento.

  - **Defectos:**

    Poca precisión, la superficie reflectante puede hacer que de valores erróneos.

    Al tener que estar como objeto tangible y siendo manipulado por el usuario, sus mediciones serían peores.

No podría estar oculto en la caja del objeto tangible dado que sus valores serían la distancia de él a la caja.

Si el diseño de la caja lo permitiese, tampoco haría medidas correctas cuando el objeto tangible rotase.

- **Sensor: Time of Flight Sensor[43]:** un sensor con un módulo que emite una seña que rebota en el destino y que permite medir las distancias con una alta precisión ignorando reflejos

  - **Ventajas:**

    Capaz de medir las distancias entre el objeto tangible del que formaría parte y el lugar de apoyo.

    Es un elemento físico muy pequeño.

    No requiere de mucha capacidad de procesamiento.

  - **Defectos:**

    Al tener que estar como objeto tangible y siendo manipulado por el usuario, sus mediciones serían peores.

    No podría estar oculto en la caja del objeto tangible dado que sus valores serían la distancia de él a la caja.

    Si el diseño de la caja lo permitiese, tampoco haría medidas correctas cuando el objeto tangible rotase.

    Sería necesario hacer uso de varios para medir todas las direcciones.

- **Sensor: Compás:** un sensor que hace uso del magnetismo para devolver información de los tres ejes.

  - **Ventajas:**

    Da información aún a través de la caja del objeto tangible.

Es un elemento físico muy pequeño.

No requiere de mucha capacidad de procesamiento.

- **Defectos:**

  El método de calibración (haciendo 8s) es más incómodo que la calibración estacionaria de los otros sensores.

  Cualquier magnetismo cercano puede variar los resultados.

- **Sensor: Infrarrojo:** un sensor que mide la proximidad y pulsos de infrarojos.

  - **Ventajas:**

    Capaz de medir las distancias entre el objeto tangible del que formaría parte y el lugar de apoyo.

    Es un elemento físico muy pequeño.

    No requiere de mucha capacidad de procesamiento.

    Bien situado podría no verse afectado por el objeto tangible.

  - **Defectos:**

    Al tener que estar como objeto tangible y siendo manipulado por el usuario, sus mediciones serían peores.

    No podría estar oculto en la caja del objeto tangible dado que sus valores serían la distancia de él a la caja.

    Sería necesario hacer uso de varios para medir todas las direcciones.

- **Ratón 3D:** es una herramienta diseñada para utilizarse en programas de desarrollo 3D como lo es CAD. Pudiendo mover los modelos en todas las direcciones usándolo junto al ratón tradicional.

  - **Ventajas:**

    Es ergonómico.

Hace más sencillo el manipular modelos 3D.

- **Defectos:**

  Tiene un alto coste.

  No tiene libertad de movimiento, tiene que estar apoyado en una superficie.

  No representa ningún objeto y es necesario manipularlo directamente.

  Trabaja de forma simultánea con el ratón tradicional.

  No se puede manipular su forma y su uso requiere entrenamiento.

2.4.2. Software

En esta sección se enumeran y describen tanto programas de creación de objetos 3D como lenguajes de programación que han sido valorados y descartados.

- **Unity**[(51)]: Es un motor de videojuegos multiplataforma, tanto de ordenadores como de smartphones, que tiene un soporte gráfico para el desarrollo de los juegos o aplicaciones y permite desarrollarlas tanto en 2D como en 3D. Su interfaz facilita mucho el desarrollo y el potente motor permite realizar ciertas tareas que serían complejas de no usar un motor gráfico. Tiene una gran comunidad de desarrolladores, aunque muchos de los assets que se realizan, son de pago. La plataforma en sí no es de pago si se utiliza para uso personal, pero en el momento de usarlo profesionalmente tiene un coste adicional. Es un motor tal vez muy pesado para el proyecto.

- **ARCore**[(18)]: Es un SDK (Software Development Kit) orientado para la realidad aumentada y desarrollado por Google para dispositivos Android e iOS. Tiene una buena variedad de APIs y sus principales capacidades son la de rastrear el movimiento, entender el entorno y la estimulación por iluminación. Al estar completamente orientado a la realidad aumentada y a los dispositivos móviles, no era útil para el proyecto.

- **AR.js**[(24)]: Un SDK para realidad aumentada basado en código abierto de JavaScript, bajo la licencia de MIT. Tiene una gran capacidad para tener unos altos fps en las

aplicaciones desarrolladas. Se centra en los sistemas de Windows Mobile, Android e iOS 11 o más. Es muy potente, pero necesita dispositivos potentes para utilizarse.

- **ARKit**[6]: Un SDK especifico para sistemas iOS. Tiene una gran potencia y esta orientado a la realidad aumentada. Al ser sólo para iOS, no era útil para el proyecto.

- **OSVR**[45]: Es una plataforma de código abierto orientada tanto a al realidad virtual como a la realidad aumentada. Soporta múltiples motores de juegos, sistemas operativos y tiene soporte para dispositivos con baja latencia de renderizado. Trabaja bajo Apache 2.0 y esta licenciado y mantenido por Sensics. Fue descartada porque actualmente se encuentra en desuso y no recibe actualizaciones.

- **Java**[19]: Lenguaje de programación de propósito general, concurrente y orientado a objetos, diseñado para tener las mínimas dependencias de implementación. Se descartó porque la interacción con clientes en un navegador requería gestionar dos lenguajes distintos, uno para el cliente, como javascript por ejemplo, y otro para el servidor.

- **C++**[48]: Lenguaje de programación basado en C que busca la orientación a objetos. Al igual que Java, para evitar duplicar los lenguajes empleados.

2.5. Tecnologías escogidas

En este apartado se describen las tecnologías escogidas para desarrollar el proyecto. Para que sea más legible, las tecnologías estarán divididas en el software y el hardware que se ha utilizado.

HITO se describe como un sistema que utiliza un objeto tangible y una interfaz en un ordenador para cumplir los objetivos descritos en la sección 1.2. El objeto tangible es un elemento que está separado físicamente del ordenador, no forma parte de él, y que tiene que contener varios sensores para leer sus movimientos. Para poder leerlos y enviarlos al ordenador, se hace uso de un microordenador con los sensores conectados.

En cuanto al software, se busca poder crear una interfaz con elementos 3D, pero que el framework o programa no sea muy costoso en memoria. Se pretende evitar el requerimiento de usar un compilador para interpretar el lenguaje y también evitar instalaciones adicionales. Además se requiere un programa para modelar los elementos 3D que tendrán la forma del objeto tangible.

2.5.1. Hardware

En esta sección se enumeran y describen los elementos físicos utilizados en el desarrollo, tanto placas como sensores y ordenadores.

- **BeagleBone** [42]

  Es un microordenador [52] creado por SeedStudio [44] y beagleboard.org [25] que permite ejecutar un sistema operativo como Linux o Android, es de bajo coste y está diseñado para tener una gran capacidad de procesamiento. Permite crear software en distintos lenguajes como Phyton, Java o C++ y es compatible con distintos sistemas operativos, como Ubuntu, Debian o Android. Tiene una entrada UART y otra Grove [41] para componentes. Este es el dispositivo elegido dado su bajo coste y que el departamento de la facultad tenía tanto este aparato como los sensores necesarios para el desarrollo del proyecto.

  En esta placa se utiliza Debian 9.9 sin entorno gráfico como sistema operativo. También ha sido probado con Debian 9.8. Escogida debido entre otras gracias a la disponibilidad de la misma, su similitud con otras placas de este tipo como Arduino y Raspberry Pi y al reto que supondría utilizarla

- **Acelerómetro**

  El sensor ADLX345 [4] es un componente con interfaz digital Grove que mide la aceleración teniendo en cuenta la gravedad de la Tierra. Mide en 3 ejes digitales y hasta 16 g. Es de bajo consumo.

- **Ventajas:**

  Tiene muy buena precisión para medir la aceleración con respecto a la de la Tierra.

  Se obtienen datos (una vez procesados) que permiten saber si están o no correctos al compararlos con la aceleración de la Tierra, evitando errores.

  Es un elemento físico muy pequeño.

  No requiere de mucha capacidad de procesamiento.

  Al ser digital no prensenta tanta sensibilidad a los ruidos.

- **Defectos:**

  Tiene que funcionar con I2C, no funciona con UART.

  Del chip utilizado concretamente, tiene una interfaz de Grove, por lo que solo es compatible con placas que tengan esta interfaz.

  Es incapaz de medir la aceleración al girarlo en sobre sí mismo en horizontal.

  Si no se controla adecuadamente puede provocar bloqueos en el ordenador.

- **Giroscopio**

El sensor ITG3200[(23)] es un sensor con interfaz digital Grove de 3 ejes de salida digital y de bajo consumo.

- **Ventajas:**

  Mide en todas las direcciones, siempre que el chip se mueve, obtiene una medición.

  Tiene una altísima precisión.

  Es muy pequeño.

  Bien controlado no requiere de mucha capacidad de procesamiento.

- **Defectos:**

Es complejo de configurar dependiendo de cómo se quiera obtener la información.

Debido a su altísima sensibilidad, en estado de reposo sigue obteniendo variaciones de los datos.

Los resultados los devuelve en grados por segundo.

Si no se controla adecuadamente puede provocar bloqueos en el ordenador.

- **Módulo Hub I2C**

  Extensión de la entrada Grove I2C para poder añadir más componentes.

- **Ordenador**

  El ordenador utilizado para el desarrollo del proyecto tiene las siguiente características.

  - **Sistema Operativo:** Ubuntu 20.04
  - **RAM:** 8 GB
  - **Procesador:** i7-8750H

  También ha sido probado con 4 GB de RAM. El proyecto se utiliza con Firefox 76.0.1 (64-bit) y está probado su funcionamiento en Chrome 84.0.4044.138 (Build Oficial)(64-bits).

2.5.2. Software

En esta sección se enumeran y describen los programas utilizados en el desarrollo.

- **NodeJS**

  Nodejs[33] es un entorno en tiempo de ejecución multiplataforma, de código abierto, para la capa del servidor (pero no limitándose a ello) basado en el lenguaje de programación JavaScript, asíncrono, con E/S de datos en una arquitectura orientada a eventos y basado en el motor V8 de Google. Fue creado con el enfoque de ser útil en

la creación de programas de red altamente escalables, como por ejemplo, servidores web. Fue creado por Ryan Dahl en 2009 y su evolución está apadrinada por la empresa Joyent, que además tiene contratado a Dahl en plantilla.

Utilizado como base para ejecutar el servidor HTTP y el API-REST para la ejecución del programa en el ordenador.

- **Express**

  Express.js[(20)] es un framework para Node.js que sirve para ayudarnos a crear aplicaciones web en menos tiempo ya que nos proporciona funcionalidades como el enrutamiento, opciones para gestionar sesiones y cookies, y otros.

  Express.js está basado en Connect, que a su vez es un framework basado en http para Node.js que se sobrepone sobre html. Podemos decir que Connect tiene todas las opciones del módulo http que viene por defecto con Node y le suma funcionalidades. A su vez, Express hace lo mismo con Connect, con lo que tenemos un framwork ligero, rápido y muy útil.

- **A-Frame**

  A-Frame[(49)]es un framework de código abierto orientado a la realidad virtual. Posee una licencia MIT, por lo que tiene una alta permisividad. Es una estructura de sistema de componente de entidad para Three.js que permite crear escenas 3D y WebVR mediante el uso de HTML. Por lo tanto, este framework funciona con JavaScript y HTML y se puede ejecutar en cualquier navegador que admita ambos lenguajes. Permite tanto modificaciones de la escena 3D como el movimiento de la cámara mediante el programa, lo que lo hace muy versátil. Además, al estar orientado a la realidad virtual, tiene soporte para controles de movimiento, el cual para este proyecto sería la BeagleBone, actuando el ordenador y este software 3D como el cliente de la placa los cuales reciben los datos.

El mayor inconveniente que implica a-frame es que es un framework que se sobrepone sobre el html, por lo que hacer uso de cualquier estructura del html queda descartada al quedar oculta y ser imposible de usar.

- **Javascript**

  JavaScript[30] es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

  Se utiliza principalmente en su forma del lado del cliente (client-side), implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas y JavaScript del lado del servidor (Server-side JavaScript o SSJS). Su uso en aplicaciones externas a la web, por ejemplo en documentos PDF, aplicaciones de escritorio (mayoritariamente widgets) es también significativo.

  Utilizado en todo el entorno del ordenador para generar el API-REST y su servidor y como base para el uso y manipulación de a-frame.

- **WebSocket**

  WebSocket[31] es una tecnología avanzada que hace posible abrir una sesión de comunicación interactiva bidireccional entre el navegador del usuario y un servidor. Se puede enviar mensajes a un servidor y recibir respuestas controladas por eventos sin tener que sondear al servidor para obtener una respuesta.

  Utilizado para la conexión entre ordenador, cliente, y BeagleBone, servidor.

- **Python**

  Python[16] es un lenguaje de programación interpretado con el objetivo primordial de ser un código legible. Es multiparadigma dado que soporta tanto orientación a objetos como programación imperativa. Al ser un lenguaje interpretado, sirve para funcionar

como base de otros programas sin necesidad de ser compilado. Es la base que se utiliza para generar el servidor del objeto tangible.

- **Tornado**

  Tornado[11] es un framework web y una libraría asíncrona creado por FriendFeed para utilizar I/O no bloqueantes. Es ampliamente escalable, permitiendo una amplia cantidad de conexiones abiertas, siendo muy apto para los sondeos de larga duración, Websockets y conexiones de larga duración. Este framework también hace uso de la librería ioloop, utilizada principalmente para generar bucles controlables.

- **Blender**

  Blender[39] es un programa informático multi plataforma, dedicado especialmente al modelado, iluminación, renderizado, animación y creación de gráficos tridimensionales. También de composición digital utilizando la técnica procesal de nodos, edición de vídeo, escultura (incluye topología dinámica) y pintura digital.

  Este programa se ha utilizado para la creación y utilización de modelados complejos en el proyecto. El problema de integrar objetos de segundas personas radica en que si no se ha creado centrando el eje de coordenadas con el centro del objeto (lo cuál no suele darse), el objeto no se centra a la hora de utilizarse como objeto tangible, por lo que no se recomendaría su uso.

- **Visual Studio Code**

  Editor de código fuente desarrollado por Microsoft para Linux, Windows e macOS con depuración, soporte de Git y capacidad para integrar plugins de sintáxis para muchos luenguajes diferentes.

  Utilizado como entorno de desarrollo para crear todo lo relacionado con el programa cliente del ordenador y el servidor de la API REST.

- **Cloud9**

  Cloud9[3] es un entorno de desarrollo integrado (IDE) basado en la nube que le permite escribir, ejecutar y depurar su código solo con un navegador. Incluye un editor de código, así como un depurador y un terminal. Cloud9 incluye herramientas esenciales para los lenguajes de programación más conocidos, como JavaScript, Python, PHP, entre otros, por lo que no necesita instalar archivos ni configurar su máquina de desarrollo para empezar nuevos proyectos.

  Cloud9 es utilizado durante el proyecto para crear el servidor que lee los sensores y espera las peticiones del cliente.

2.6. Conclusiones del estado del arte

Durante todo el estado del arte, se han podido investigar artículos y plataformas que tienen relación con los objetos tangibles. De todo lo investigado hasta este punto, se puede deducir que se necesita un objeto físico que haga de objeto tangible y que pueda ser manipulado y, de forma simultánea cuando es manipulado, que el movimiento que se hagan sobre este objeto físico se vea reflejado en una copia en la pantalla del ordenador para mejorar la sensación de profundidad en el espacio tridimensional. La copia tiene que ser igual al objeto real para poder crear una relación ojo-mano y tiene que poder cambiar de forma provocar una mayor inmersión. Además, la interfaz que se muestre en el ordenador donde se vea la copia, tiene que ser sencilla y amigable con el usuario.

Se puede ver cómo existen herramientas y plataformas que pueden ayudar a las personas, aunque dichas herramientas están pensadas más para los niños. Dan a entender algunos conceptos básicos como las formas y tamaños de los objetos, cómo algunas tienen una representación digital en la pantalla y cómo otras hacen de los objetos tangibles, un apoyo directo para realizar acciones en la pantalla. Las ideas son muy similares a las que se plantean

en el proyecto de esta memoria. El único problema de las mismas es el coste y que no están totalmente dirigidas al problema que se quiere tratar, entender las representaciones de los objetos físicos en pantalla como tridimensionales y los espacios tridimensionales.

De las tecnología descartadas se puede obtener que hay bastante variedad de elementos que pueden ayudar con la realización del proyecto, pero que pueden no servir para el propósito del mismo.

Con las tecnologías escogidas se puede empezar a concretar el proyecto y cómo puede ser desarrollado.

Teniendo en cuenta todo lo expuesto, el objeto tangible debe de poder enviar la información al ser manipulado al ordenador. Para ello, debe poder controlar los sensores que permitan leer los movimientos y enviarlos al ordenador, por lo tanto, tiene que ser un microordenador de bajo coste. En cuanto a la interfaz, no debe ser compleja y debe contener los elementos necesarios para que la representación pueda cambiar de forma a la del objeto tangible.

Capítulo 3

Requisitos del sistema

En este capítulo se van a concretar los requisitos del proyecto, su interfaz y las tecnologías utilizadas. Para ello primero se hará una investigación de los usuarios objetivo del mismo, creando un conjunto de personas ficticias y, desde estas, los requisitos que ha de cumplir el sistema. Además se concretará la interfaz para los usuarios y la carcasa que cubrirá que dará forma al objeto tangible.

Para conseguirlo, se busca a partir de pequeños puntos, obtener el diseño final de una aplicación o sistema. Primero se plantea una investigación del usuario para encontrar los tipos de usuario objetivo a partir de la premisa del proyecto. Después, con esos usuarios, se modelan personas ficticias para poder obtener los objetivos del sistema de forma más definida. Luego se obtendrán los requisitos a partir de toda la información previa para describir el producto en el siguiente punto junto con los casos de uso. Como punto final, se seleccionarán y descartarán las tecnologías que usará HITO.

3.1. Investigación del usuario

El objetivo final de esta investigación es reflexionar sobre qué tipo de usuario o usuarios se pueden beneficiar del sistema. Para ello, se ha realizado una hipótesis de los posibles usuarios.

3.1.1. Hipótesis de persona

Se ha planteado qué personas podrían tener dificultad en entender figuras tridimensionales en pantallas y problemas de orientación espacial en entornos digitales tridimensionales. Este proyecto no está dirigido completamente a los niños, pero parte de los usuarios que pueden ser objetivo son ellos. El problema del entendimiento puede aparecer en personas de cualquier edad que no hayan interaccionado con elementos 3D.

Otro tipo de usuarios que se puede plantear son aquéllas personas, tanto niños como adultos, que presentan problemas de logopedia de orientación espacial, es decir, trastornos que dificultan que se orienten en el espacio, más aún en entornos tridimensionales debido a la falta de referencias físicas.

De forma adicional, personas con problemas médicos en la vista pueden presentar problemas para entender la profundidad de los objetos al no tener ambos ojos correctamente alineados o con diferencias de visibilidad entre ellos.

Se entiende, por lo tanto, que los usuarios se diferencian en:

- Niños y adultos que están iniciando su recorrido en el mundo tecnológico, más concretamente, en el relacionado con entornos tridimensionales como pueden ser los videojuegos o aplicaciones de realidad virtual/aumentada.
- Niños y adultos con problemas de logopedia.
- Niños y adultos con problemas de visión como estrabismo, con diferencias de visibilidad entre ambos ojos o que no desarrollen la estereopsis[53].

Todos estos usuarios nombrados anteriormente se pueden agrupar en un solo tipo de usuario dado que todos estos problemas tienen en común la falta de entendimiento de los entornos tridimensionales.

3.2. Modelado

En este apartado se busca una descripción imaginaria que represente y encapsule la información sobre los usuarios objetivo del sistema de la sección 3.1. De esta manera, se centra el estudio en el usuario objetivo, usuarios de causa logopédica, aquel que tiene problemas del entendimiento tridimensional.

Para crear estas personas se va a utilizar un proceso top-down, propuesto en el libro 'The Persona Lifecycle'[36]. Se van a seguir los siguientes pasos:

- Identificar las categorías de los usuarios
- Procesar los datos
- Identificar y crear los esqueletos

3.2.1. Identificar categorías de usuarios

En esta parte se deben crear las categorías de los usuarios para encajar la información extraída en anteriores secciones de la memoria. Para categorizarlos, se ha decidido separarlos atendiendo a la causa que les provoca el problema. Se pueden desglosar en usuarios de 'Causa logopédica', donde se incluyen aquéllos usuarios que empiezan con el uso de la tecnología entorno a los espacios tridimensionales y en usuarios de 'Causa visual' donde se incluyen todos aquéllos con problemas de visión. Además, se añade un usuario externo, sin causas ni problemas conocidos pero al que le interesa la interacción entre objetos en el entorno virtual y el objeto físico. Este tipo de usuario no está relacionado con el objetivo principal del proyecto y simplemente buscan otra forma de utilizar el mismo.

Para utilizar un nombre más apropiado, los usuarios de 'Causa logopédica' se nombrarán como 'Iniciados' y aquéllos de 'Causa visual' se nombrarán como 'Vista'. Los usuarios sin causa o sin problemas que también quieran hacer uso del sistema, van a categorizarse como usuarios 'Externos'. Los tipos de usuarios finales serían:

- **Iniciados:** Todos los usuarios que comienzan a usar la tecnología de entornos tridimensionales y aquéllos con problemas de logopedia.
- **Vista:** Todos los usuarios con algún tipo de problema de visión que dificulte el acercamiento a los entornos tridimensionales.
- **Externos:** Todos los usuarios sin ningún tipo de problema relacionado con los objetivos principales del sistema y que quieren utilizarlo.

3.2.2. Procesar los datos

En este punto se debe extraer toda la información relevante de los apartados anteriores para aplicarlos al usuario objetivo del sistema. Los puntos de la sección 2.3 generan junto a los tipos de usuario objetivo un grupo de puntos importantes del usuario, reduciendo muchos de los puntos anteriores al objeto físico con la representación en la interfaz.

Iniciados	Buscan entender entre la profundidad en pantallas planas Necesitan saber la forma del objeto mostrado en el ordenador Necesitan entender cómo se mueven los objetos en el ordenador
Vista	Buscan ver bien la forma del objeto físico en el ordenador Buscan entender la profundidad en pantallas planas.
Externos	Buscan utilizar el sistema para otro propósito

Cuadro 3.1: Tabla de puntos extraidos de usuario

3.2.3. Identificar y creación de esqueletos

Después de haber encontrado qué tipos de usuarios tiene el sistema, hay que identificar cada uno de ellos y crear un esqueleto para cada tipo. El resultado ha sido la obtención de dos tipos de usuarios claramente definidos, los usuarios objetivos y los no objetivos que usan el sistema. Como el tipo de usuarios relacionado con el sistema solo es un tipo, Iniciados, no hay necesidad de identificar y separar los esqueletos, dado que los usuarios Externos no cumplen ningún requisito más allá de querer usar el sistema. Con el tipo Iniciados se tiene un

subtipo de usuarios y se va a proceder a crear dos esqueletos, uno para iniciados y otro con algún problema de los de iniciados en adición con su problema de visión. La prioridad de los usuarios es poder solucionar o reducir su problema respecto del entendimiento tridimensional en ordenadores y así se mostrará en los esqueletos que se van a crear en el siguiente apartado. En adición, se va a crear un esqueleto para el tipo Externos.

3.2.4. Desarrollo de personas según esqueletos

Teniendo en cuenta la prioridad de los esqueletos mencionada en el apartado anterior, se va dar personalidad y profundidad a los tipos mencionados. Uno de ellos será Pablo Sánchez, un niño pequeño que está comenzando a utilizar la tecnología y del que no se está seguro si tiene algún problema de logopedia; el segundo será Pedro Casado, un joven adulto que tiene un alto estrabismo en uno de sus ojos y le causa problemas de profundidad, sobre todo cuando se trata de mirar en una pantalla un entorno tridimensional, y el tercero Sara López, un estudiante universitario que está interesado en utilizar los objetos tangibles para crear interacciones entre los objetos del entorno virtual, ya sea para generar animaciones, diálogos u otros objetivos.

Pablo Sánchez

- **Nombre:** Pablo Sánchez
- **Edad:** 10
- **Nivel de estudios:** Estudiante de primaria
- **Breve descripción:** Un pequeño estudiante que tiene problemas cuando intenta jugar a algún juego 3D que le han mostrado sus padres

- **Objetivos y motivaciones:** Divertirse, divertirse jugando juegos y mejorar mucho para no pasarlo mal
- **Conocimientos tecnológicos:** Ninguno o casi ninguno, apenas toca la tecnología excepto cuando intenta jugar con su padre
- **Detalles:** Pablo no tiene ninguna rutina en su día a día, casi todo es esporádico y es algo que le gusta.

  Va al colegio y hace sus tareas con tranquilidad, no es un mal estudiante y quiere llegar a ser como su madre, una arquitecta de videojuegos, aunque en realidad Pablo no conoce esa profesión, solo ve a su madre moviendo algo en la pantalla que parecen fotos de algunos objetos de la casa.

  Aunque le gusta eso, tiene el problema que cuando intenta ver que hace su madre y esta trabaja mientras él la ve, no es capaz de entender lo que está haciendo incluso cuando su madre le enseña un cojín de la casa y la copia que ha hecho ella., parece que no tiene la suficiente inmersión de esta forma.

  Su padre intenta jugar con él a algunos juegos infantiles en tres dimensiones en la televisión, pero falla mucho y no comprende los objetos del juego.

  Piensa que estaría bien poder hacer que si ese cojín que tiene en casa y el que hizo su madre se moviesen a la vez, tal vez entendería mejor ese cojín virtual.

Pedro Casado

- **Nombre:** Pedro Casado
- **Edad:** 24
- **Nivel de estudios:** Universitarios
- **Breve descripción:** Es una persona alegre que tiene ciertos problemas para notar la profundidad debido a un problema de vista. En la realidad ya se ha acostumbrado, pero en lo digital no.
- **Objetivos y motivaciones:** No tener el miedo de fallar en un trabajo por su problema y poder trabajar con modelos 3D.
- **Conocimientos tecnológicos:** Usa todo lo básico de los smartphones u ordenadores, pero cuando se trata de juegos en 3D o trabajos que requieran herramientas en 3D no ha tocado mucho en su vida.
- **Detalles:** Pedro trabaja en una fábrica de vehículos como ingeniero de máquinas. Su trabajo es asegurar que las piezas físicas salen correctamente, para lo hace uso de unos planos impresos en caso de tener que hacer medidas. Vive con el temor de tener que usar la pantalla del ordenador con el modelo en 3D de la pieza para comprobar que está correctamente.

  Alguna vez le ha tenido que tocar hacer eso, pero por suerte, su amigo y compañero de trabajo le ayuda en esa faena.

  Quiere poder hacerlo por su cuenta y no depender de su amigo, pero su estrabismo le impide hacerlo. No quiere operarse por miedo y para él, esa no es una solución.

Al igual que hace con los objetos reales, que los palpa para comprobar su profundidad y así saber su tamaño exacto, le gustaría poder hacerlo con los modelos en 3D, pero eso es algo que está fuera del alcance de sus manos.

Sara López

- **Nombre:** Sara López
- **Edad:** 21
- **Nivel de estudios:** Bachillerato
- **Breve descripción:** Es una joven estudiosa que está cursando una Ingeniería en Informática. No tiene demasiados problemas para avanzar por los cursos y es bastante curiosa con los temas que relacionan hardware con software. Tampoco hay que descartar que le encantan los videojuegos.
- **Objetivos y motivaciones:** Como estudiante, quiere terminar la carrera. En lo personal, ha encontrado una pequeña afición con el uso de placas de bajo coste.
- **Conocimientos tecnológicos:** Conocimientos altos en lo relacionado con la informática.
- **Detalles:** Actualmente Sara no trabaja, estudia su carrera y sale con los amigos siempre que puede. Su día a día no se ciñe por una rutina fija a excepción de ir a clases. Le gusta tanto montar y desmontar ordenadores como la creación de programas más allá de los obligados por la carrera.

Hace poco tiempo descubrió, gracias a la carrera, el uso de placas de bajo coste como las placas Arduino, Raspberry o las Beaglebone y es un mundo que le gusta. Quiere investigar todo lo posible formas de interactuar con estas placas de forma similar a como se hace con las nuevas tecnologías de realidad virtual, usar objetos físicos que se relacionan intrínsecamente con objetos virtuales.

3.2.5. Validación de las personas

Se puede apreciar que en los primeros esqueletos se intenta ahondar con el problema encontrado en fases anteriores, por lo que estos dos esqueletos incurren en los resultados obtenidos anteriormente y podría decirse que ambos son válidos. El tercero es el caso especial que no tiene que ver con los resultados anteriores relacionados con el objetivo principal, sino con el secundario, el cual cumple y podría decirse que el tercer esqueleto también es válido.

3.3. Requisitos

Usando los actores desarrollados, se definirán casos de uso para determinar qué funciones desarrollar.

En esta fase a través del desarrollo de escenarios, una descripción de las tareas que realiza una persona para conseguir diferentes objetivos, identificaremos los requisitos que necesitará tener nuestra aplicación para determinar las necesidades de las personas.

Para lograr la definición de los requisitos, se han seguido cuatro etapas.

3.3.1. Enunciado de problemas y visiones

Se tienen varios enunciados de **problemas** que especifican una situación que ha de cambiar y para cada uno, se tendrá una **visión** para invertir el problema.

- **Problema:** Pedro no entiende que es posible que algún objeto real esté dentro de una pantalla de ordenador.

  Visión: El sistema consistirá en tener un objeto digital tridimensional que imita un objeto físico, en este caso el tangible object.

- **Problema:** Pablo quiere poder desplazar el modelo 3D con un objeto físico exactamente igual a este en el mundo real para aumentar la inmersión.

  Visión: El sistema detectará los desplazamientos del tangible object para que su representación los imite.

- **Problema:** Pablo también necesita rotar el objeto para ver que son el mismo objeto.

  Visión: El sistema detectará las rotaciones del tangible object para que su representación los imite.

- **Problema:** Pedro necesita que se puedan añadir nuevos modelos para el objeto y elegir entre ellos y que el objeto tangible pueda cambiarse para que tenga la misma forma.

  Visión: Se añadirá la función de poder elegir el modelo y alguna forma de añadir modelos. También se utilizará una carcasa o cobertura para el objeto físico para que cambie de forma.

- **Problema:** Pedro necesita referencias para saber la profundidad de los objetos 3D en la pantalla.

  Visión: Además de la representación del tangible object existirán otros objetos de referencia, estos sólo en su modelos 3D, para comprobar que los desplazamientos se realizan y que sirvan de referencia.

- **Problema:** Tanto Pablo como Pedro necesitan saber que parte del modelo están viendo en cada momento.

Visión: Teniendo en cuenta este punto, el modelo tendrá que tener en cada cara un distintivo, como colores o letras.

- **Problema:** Sara quiere probar a interaccionar con objetos mediante el uso de un objeto tangible, de una placa física. También quiere poder añadir más funcionalidades.

  Visión: Además de lo expuesto anteriormente, cuando la representación del objeto tangible toque un objeto de la escena, ocurrirá un cambio para demostrar la interacción. Además deberá haber alguna forma de añadir nuevas funcionalidades.

3.3.2. Expectativas de los usuarios

En este punto, se va a hablar de las expectativas, que son las situaciones en las que las personas creadas en la sección 3.2 que representan los posibles requisitos que el sistema ha de cumplir como mínimo.

Pablo Sánchez

Después de ir al colegio y, si su madre se encuentra en casa, le gusta observar cómo hace objetos realistas en una pantalla. Aunque sigue sin poder entender bien la relación entre ese objeto físico y el digital.

En ocasiones su madre le deja mover el modelo con el ratón, pero claro, no lo siente como si fuese dicho objeto. Aunque ella tampoco sabe qué es lo que no entiende bien y Pablo tampoco sabe expresarlo adecuadamente.

Casi siempre, aunque no todos los días, su padre empieza a jugar a algún juego en el televisor. Generalmente son juegos simples como un plataformas, pero la mayoría en 3D. Pablo se sienta junto a él e intentan jugar juntos, pero Pablo siempre falla porque no relaciona que con un botón de un mando con una forma concreta se pueda hacer mover al personaje.

Necesita algo con lo que aprender esa correlación.

Pedro Casado

Como todos los días, se va a trabajar con la angustia de que su compañero no vaya al trabajo y tenga que solucionar algún problema de pantalla relacionado con las piezas él solo, pues el trabajo pide rapidez.

Dependiendo de la jornada, tiene que comprobar físicamente la pieza, para lo cual no tiene mucho problema. De vez en cuando y con su compañero en los ratos libres, utilizan la máquina para comprobar los modelos de las piezas en 3D. Su amigo le intenta enseñar como puede, aunque ni él mismo comprende muy bien su problema con la profundidad.

Hablando de ello con Pedro, este le comenta que no es sólo la profundidad lo que le falla, sino también algunos aspectos del modelo. A veces no sabe qué cara es cada una en una figura. No comprende bien el tamaño debido a que sólo aparece la pieza representada sin muchos más datos ni referencias. También le gustaría poder mover la pieza y que el modelo se moviese, pero por lo general, las piezas son muy grandes.

Sara López

Como estudiante universitaria activa, tiene que ir a clases prácticamente todos los días de la semana, exceptuando los fines de semana. En los fin de semana aprovecha para investigar lo que ella considera su nueva afición, investigar sobre placas de bajo coste y los objetos tangibles.

Quiere ver y conocer cómo interactúan los objetos físicos con su representación virtual junto con otros objetos virtuales.

3.3.3. Escenarios de contexto

En este apartado se mostrarán las acciones que permite el sistema, mostrando cómo soluciona los problemas. Para ello se reescribe la historia destacando el nuevo elemento.

Pablo Sánchez: Como ya se sabe de Pablo, su rutina no es muy rutinaria más allá del colegio. Cuando está en casa y su madre está modelando, siempre se suele acercar a observar. Ahora ya no es como antes, ahora posee HITO. Su madre ha modelado la forma del objeto tangible, en esta ocasión un simple cubo alargado, en el ordenador y le da el tangible object de HITO a su hijo. Cuando Pablo mueve el objeto, el modelo del ordenador hace las mismas acciones, si lo gira, el modelo lo hace, si lo mueve, lo mismo. Esto le ayuda a entender un poco mejor la relación entre objeto físico y digital. En el caso de su padre, el sistema no le ayuda mucho, pero como ha practicado con su madre, empieza a entender que ese mando con botones son cómo el objeto físico que mueve al digital.

Pedro Casado: Pedro como siempre, va a trabajar, aunque ahora menos nervioso, dado que tiene el sistema con algunas formas de las piezas de los vehículos. Cuando ocurre algún problema con alguna pieza, su amigo sigue estando con él, pero ahora gracias a HITO, que se ha integrado en el ordenador que contiene los modelos de las piezas, tiene menos dificultades. Cuando una puerta, por ejemplo, tiene un error y hay que comprobar su modelo, Pedro cambia la forma del tangible object a a la de una puerta y en el programa se muestra el modelo de la misma con los lados de distintos colores. Con todo esto, Pedro puede hacer su trabajo con menor dificultad.

Sara López: Ya se conoce que Sara solo usaría HITO para aprender y ver cómo interaccionan los objetos entre sí. HITO no le ofrece un tutorial de cómo generar un modelado de un objeto real, pero sí le ofrece vías de ver las interacciones entre el objeto físico y el modelo y los otros objetos virtuales. También cubre la necesidad de poder ampliar la funcionalidad.

3.3.4. Identificación de requisitos

Los requisitos del sistema se deducen a partir de las personas modeladas en la sección 3.2.4 y las secciones 3.3.1 y 3.3.2. Los requisitos son los siguientes:

- **Tiene que haber un objeto físico y un modelo tridimensional del mismo** como se menciona en los problemas de la sección 3.3.1.
- **Se tiene que poder desplazar el modelo 3D en todos los ejes cuando el objeto físico lo hace** como se menciona en los problemas de la sección 3.3.1.
- **Se tiene que poder rotar el modelo 3D en todos los ejes cuando el objeto físico lo hace** como se menciona en los problemas de la sección 3.3.1.
- **Han de poder verse más objetos en la escena a parte del que representa el objeto físico** como se menciona en los problemas de la sección 3.3.1.
- **El modelo ha de estar diferenciado en sus caras, cambiándolas de color o teniendo iluminación** como se menciona en los problemas de la sección 3.3.1.
- **Se tienen que poder añadir nuevos modelos** como se menciona en los problemas de la sección 3.3.1.
- **El modelo físico debe poder tener varias formas así como lo debe tener el modelo 3D** como se menciona en los problemas de la sección 3.3.1.
- **La representación del objeto físico ha de poder interactuar de alguna forma con el resto de objetos modelados en el entorno virtual** como se menciona en los problemas de la sección 3.3.1.
- **El sistema debe ofrecer una facilidad de añadir nuevas funcionalidades sin demasiado esfuerzo** como se menciona en los problemas de la sección 3.3.1.

3.4. Descripción del producto

En este apartado se define con mayor precisión lo que se espera del producto en la parte que hace uso del software 3D con los modelados. Esta fase pasará por varias iteraciones hasta llegar al boceto final que cumpla los requisitos anteriores.

El primer sub apartado es 'Definir el factor forma, la postura y los métodos de entrada' en el que se investigará cómo presentar la información, la atención que debe tener el usuario cuando se use el sistema y las formas de interactuar con el mismo.

Después se definirán los elementos que tendrá la interfaz hasta el punto de llegar a la utilización del objeto tangible.

Se obtendrán los grupos funcionales para esquematizar el sistema para después crear la interfaz relacionada con el software 3D mediante prototipos a papel o herramientas de diseño de prototipos.

Para verificar los prototipos se crearán escenarios *key path* que describen cómo interactúa la persona, en este caso la persona modelada anteriormente, con la interfaz diseñada y validar los diseños con los escenarios de validación para poder encontrar fallos en el sistema y ajustarla para cubrir las necesidades de los usuarios.

Una vez validados estos escenarios, se detallará en profundidad el flujo de la interfaz, apoyándose en los casos de uso que reflejen cada acción del flujo.

3.4.1. Definir el factor de forma, la postura y los métodos de entrada

El factor de forma de HITO se define como un programa para ordenadores, preferiblemente un portátil para permitir mayor movilidad y no encontrarse el proyecto en un lugar fijo como sería un sobremesa. También es necesario hacer uso de un elemento físico como

objeto tangible.

En cuanto a la postura, depende de quién lo esté utilizando. Si el usuario es del tipo **Iniciados**, será un sistema temporal, haciendo él uso del mismo o mediante un logopeda que esté utilizando la aplicación para resolver los problemas del usuario; Para el tipo **Vista** pasaría lo mismo que con el anterior tipo. Si el usuario es del tipo **Externos**, puede que pase del plano temporal, dependiendo de para qué decida utilizar el sistema.

Como métodos de entrada, se definen dos. El primero es mediante el ratón del ordenador, para interactuar con la interfaz hasta alcanzar el punto en el que entra el segundo método de entrada: el objeto tangible. Con el objeto tangible se moverá su representación en la interfaz al igual que se hace con el objeto real, por lo que requerirá de un diseño de objeto.

3.4.2. Definir los elementos de datos y funciones

Para definir los elementos funcionales, definimos primero como elementos de datos las posibles acciones dentro de la interfaz del sistema. Estas posibles acciones se obtienen a partir de desgranar los requisitos identificados en la sección 3.3.4. Las posibles acciones son las siguientes:

- Diferenciar las distintas acciones del tangible object.
- Rotar el modelo del objeto tangible.
- Desplazar el modelo del objeto tangible.
- Cambiar la forma del modelo del objeto tangible.
- Ver las posibles distintas formas del modelo del objeto tangible.
- Añadir nuevas formas no preestablecidas.
- Elegir la escena de rotar o desplazar.

- Menú necesario para desplazarse entre las distintas acciones.

Una vez definidos los elementos de datos y utilizando los escenarios de contexto y requisitos de las personas, se crean los siguientes elementos funcionales:

- Pablo puede desplazar el objeto modelado con el objeto tangible.
- Pedro puede rotar el objeto tangible y verlo reflejado en la pantalla.
- Pedro puede cambiar la forma del modelo para que coincida con lo que necesita.
- Pedro puede utilizar una impresora 3D para cambiar la forma del objeto tangible.
- Sara puede hacer interactuar el objeto tangible con distintos objetos dentro de la interfaz.

3.4.3. Determinar los grupos funcionales y las jerarquías

Se agrupan los elementos en unidades funcionales, separándose por distintas vistas que contengan las funciones agrupadas de forma lógica. Para facilitar el entendimiento de qué hace cada 'escena', se le añade un nombre sencillo:

- Menú Principal [Rotar, Mover, Libre, Modificar [Actualizar forma, Seleccionar forma]]

3.4.4. Prototipado iterativo

En esta sección se van a ver los prototipos de HITO basándose en los requisitos identificados en la sección 3.3.4 y apoyándose en la sección 3.4.2. Antes de pasar a su versión final en el ordenador. Se expondrá inicialmente la idea del proyecto y luego prototipos a papel de la interfaz. También se añaden los bocetos de la carcasa del objeto tangible para representar un objeto concreto que se visualizaría en la interfaz.

El objetivo de este prototipo es acercar una idea de cómo terminará siendo tanto la interfaz como el objeto físico, evitando crear muchas versiones distintas y centrándose en solo uno.

- **Concepto inicial**

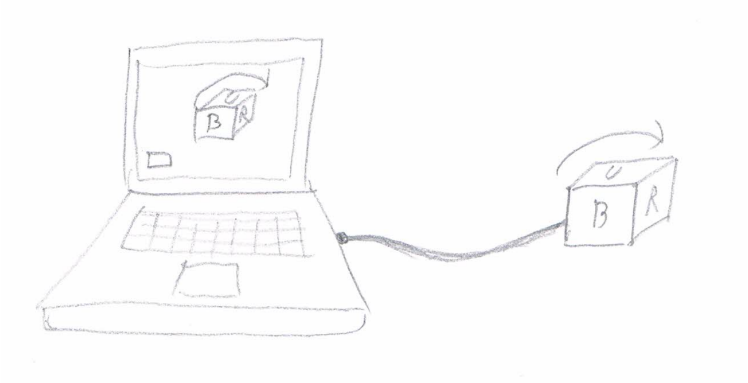


Figura 3.1: Concepto del sistema

El concepto inicial de HITO era hacer uso de un objeto tangible, a la derecha de la imagen, junto con un ordenador, a la izquierda. A lo largo del capítulo 2, se ha expuesto que es necesario utilizar tanto un ordenador como un objeto tangible que, para ser manipulado, tiene que estar separado físicamente del ordenador.

Primer boceto de la interfaz

En los primeros bocetos, como se puede apreciar en la figura 3.2, se diseña una interfaz simple basándose en los requisitos identificados en la sección 3.3.4 y en la tabla 2.1 en la que siempre suele ser una interfaz sencilla con unos pocos botones.

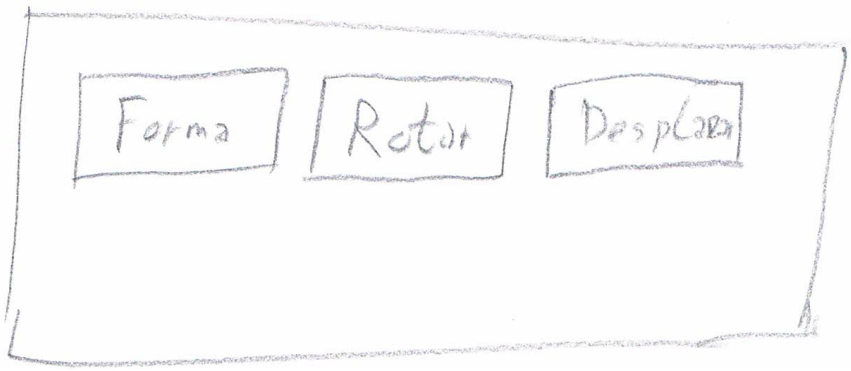


Figura 3.2: Primer prototipo del menú

En la figura 3.3, se presenta como se vería la escena de cambiar el objeto como los requisitos identificados en la sección 3.3.1, estando sin decidir aún algunas de las posiciones o la forma. Además, se plantea la posibilidad de crear tus propios modelados desde el programa.

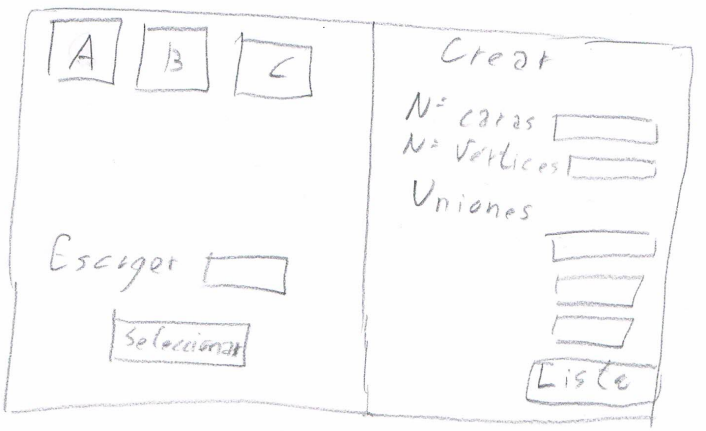


Figura 3.3: A la izquierda la selección de la forma; A la derecha la creación de un objeto

En la siguiente figura se pueden apreciar las dos vistas que se tenían planteadas inicialmente para la utilización del objeto tangible con su representación, teniendo en una la rotación del objeto y en la otra el desplazamiento como en los requisitos de la sección 3.3.1, teniendo en un principio sólo dos opciones, sin plantearse el combinar ambas vistas.

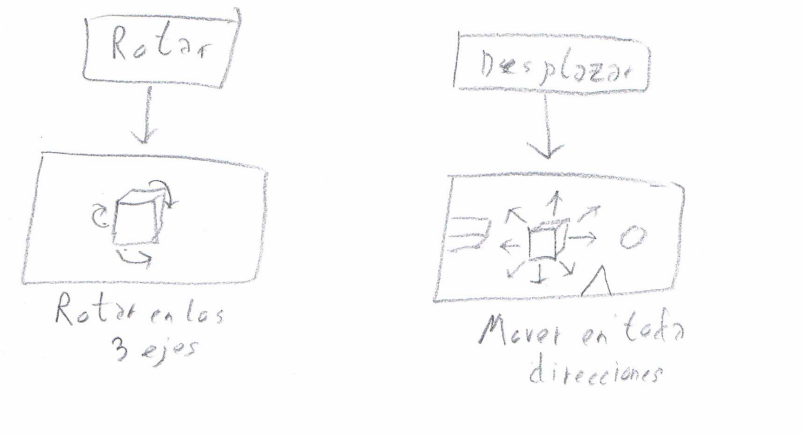


Figura 3.4: Primer prototipo de Rotar y Desplazar

- **Segundo boceto de la interfaz**

Este segundo boceto se plantea después de estudiar nuevamente los requisitos y las ideas planteadas. También ocurre después de investigar los distintos software 3D para crear la interfaz directamente utilizando modelos tridimensionales.

En esta nueva figura 3.5, se aprecia cómo se ha planteado una interfaz más detallada y acorde con el sistema, teniendo las cuatro escenas planteadas en un principio. Además se añade una interacción más para entrar en esta escena con las otras cuatro. Se añade como punto de integración un botón para conectar y otro para desconectar del objeto tangible para casos en los que pueda haber problemas con el mismo.

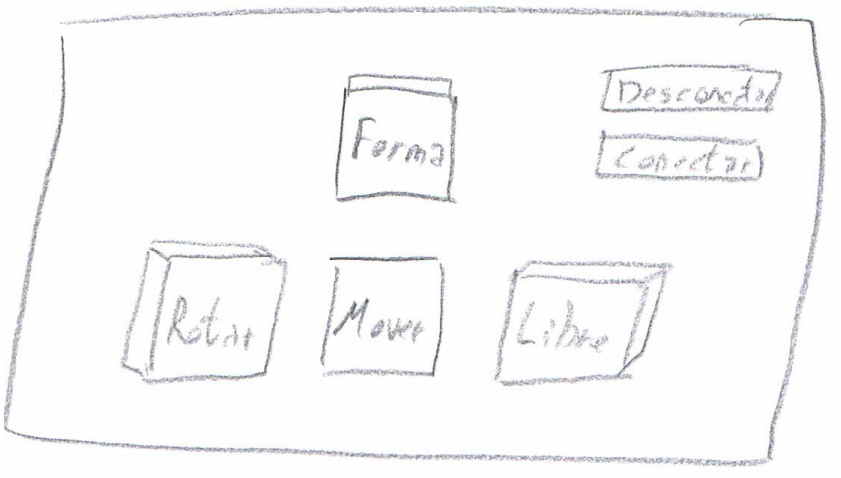


Figura 3.5: Segundo prototipo de menú

En la siguiente figura 3.6 se aprecia el cambio dado a la selección de las distintas formas de la representación del objeto tangible. También se aprecia la eliminación de la sección de creación de objetos y se sustituye por un 'botón'.

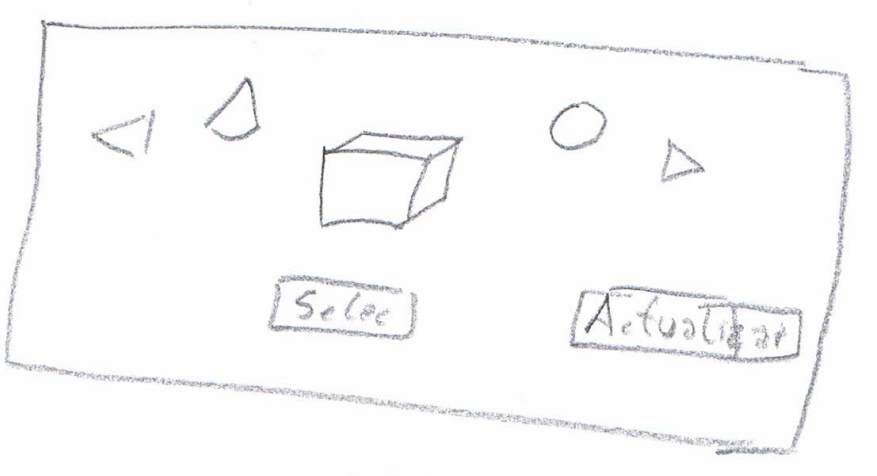


Figura 3.6: Segundo prototipo del menú de objetos

En esta última figura 3.7 del prototipo, se pueden ver las tres secciones definidas como Rotar, Mover y Libre. En las dos últimas vemos cómo se han añadido más objetos al escenario. También se añade un botón para salir de esas escenas y volver al menú con los botones de las cuatro.

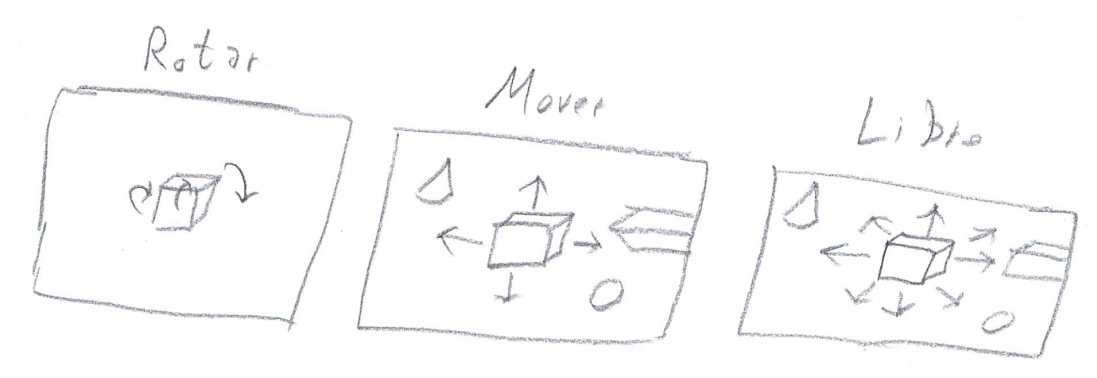


Figura 3.7: Segundo prototipo de Rotar, Mover y Libre

- **Primer boceto de la carcasa**

Después de plantearse el cambio de forma del objeto físico en los requisitos de la sección 3.3.4, se plantea un prototipo de forma. En el primer boceto de la carcasa, como se puede ver en la figura 3.8, se planeó utilizar un cubo como primer elemento.

que luego se pasaría a una versión hecha con cartón.

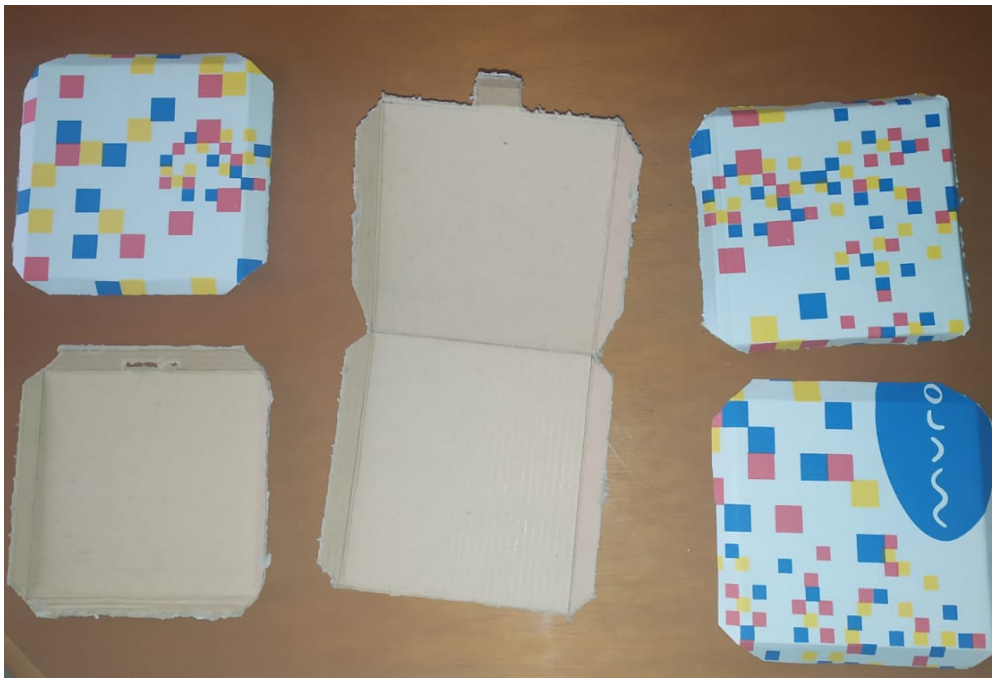


Figura 3.8: Primer prototipo carcasa

El resultado podría verse muy engorroso, incluso alguien con las manos grandes podría tener dificultad al usarlo. Además, los sensores tendrían libre movimiento dentro del espacio de la caja, lo que supondría lecturas incorrectas.

En esta ocasión se plantea hacer un cubo alargado, reduciendo mucho su tamaño. Se añade una nueva pestaña cerca de la tapa donde van los sensores colocados, pero con el inconveniente de estar boca abajo y mirando hacia atrás. También se añade un agujero para el cable de corriente, otro en el lateral para acceder a los botones del aparato y una rendija en la parte inferior para que no se sobrecaliente. En la figura 3.9 se puede apreciar la carcasa desmontada y en la figura 3.10 la misma abierta y cerrada.

Figura 3.9: Carcasa

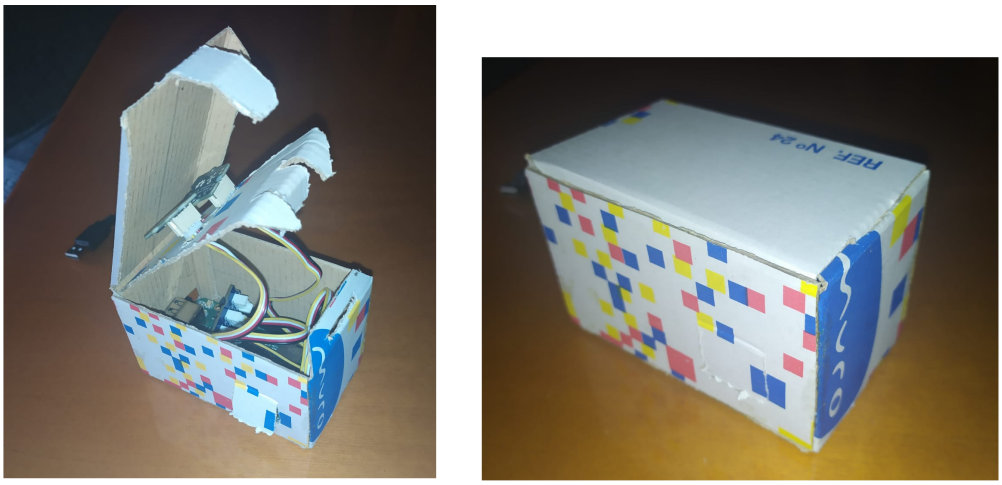


Figura 3.10: Objeto tangible

3.4.5. Casos de uso

Con la información previa se procede a mostrar los casos de uso de la interfaz. Se presupone una condición innata a todos, que el sistema se está ejecutando, tanto la parte de la interfaz como el servidor de la BeagleBone y que ambas partes ya están abiertas.

3.2: Caso de uso - Entrar en el menú

Descripción	Debe cambiar a la pantalla del menú
Actor	Usuario
Precondiciones	Estar en la pantalla del botón Start
Postcondiciones	Mostrar el menú con los cuatro cubos
Flujo	El usuario pulsa el botón Start y la pantalla cambia al menú con los cuatro cubos.

3.3: Caso de uso - Desconectar del objeto tangible

Descripción	Debe de desconectarse del servidor del objeto tangible
Actor	Usuario
Precondiciones	Estar en el menú principal
Postcondiciones	Mostrar el botón reintentar y desconectarse
Flujo	El usuario pulsa el botón desconectar, se desconecta del servidor

3.4: Caso de uso - Conectar el objeto tangible

Descripción	Debe conectarse con el servidor del objeto tangible
Actor	Usuario
Precondiciones	Estar en el menú principal
Postcondiciones	Mostrar el botón Desconectar y conectarse
Flujo	El usuario pulsa el botón Reintentar, se conecta al servidor

3.5: Caso de uso - Intentar entrar en Rotar, Mover o Libre

Descripción	No debe poderse entrar en Rotar, Mover o Libre
Actor	Usuario
Precondiciones	Estar desconectado del objeto tangible
Postcondiciones	No cambiar de pantalla
Flujo	El usuario pulsa Rotar, Mover o Libre y no debe realizarse ninguna acción

3.6: Caso de uso - Entrar en Forma

Descripción	Debe cambiarse de pantalla a la de Forma
Actor	Usuario
Precondiciones	Estar en el menú principal Estar conectado o desconectado del objeto tangible
Postcondiciones	Mostrar la pantalla de Forma
Flujo	El usuario pulsa en Forma y la pantalla cambia a la de Forma

3.7: Caso de uso - Rotar carrusel

Descripción	Deben cambiar los objetos mostrados en el carrusel del centro de la pantalla
Actor	Usuario
Precondiciones	Estar en la pantalla de Forma
Postcondiciones	Cambiar los objetos del carrusel
Flujo	El usuario pulsa en la flecha de la izquierda o de la derecha y el carrusel cambia

3.8: Caso de uso - Actualizar carrusel

Descripción	Debe actualizarse el carrusel
Actor	Usuario
Precondiciones	Estar en la pantalla de Forma
Postcondiciones	Debe añadirse una nueva forma al carrusel Debe eliminarse una nueva forma del carrusel Debe actualizarse el carrusel pero no cambiar nada
Flujo	El usuario pulsa el botón Actualizar y el carrusel se actualiza dependiendo de si se han añadido o eliminado objetos en 'assets/models'

3.9: Caso de uso - Rotar el objeto

Descripción	Debe mantenerse pulsado el objeto central del carrusel y mover el ratón para hacerlo rotar
Actor	Usuario
Precondiciones	Estar en la pantalla de Forma
Postcondiciones	Debe rotarse el objeto central del carrusel
Flujo	El usuario mantiene pulsado el objeto central del carrusel y mueve el ratón haciendo rotar al objeto

3.10: Caso de uso - Salir de la pantalla de Forma

Descripción	Debe volver a la pantalla del menú principal
Actor	Usuario
Precondiciones	Estar en la pantalla de Forma
Postcondiciones	Mostrar el menú principal
Flujo	El usuario pulsa el botón Back de la pantalla

3.11: Caso de uso - Cambiar forma al objeto tangible

Descripción	Debe volverse al menú principal cambiando la forma de los cubos del menú y del objeto tangible
Actor	Usuario
Precondiciones	Estar en la pantalla de Forma Cambiar a una forma distinta
Postcondiciones	Mostrar el menú principal sin cambios Mostrar el menú principal cambiando la forma de los cubos
Flujo	El usuario, habiendo rotado el carrusel o no, pulsa el botón Seleccionar

3.12: Caso de uso - Entrar en Rotar

Descripción	Debe cambiar a la pantalla de Rotar
Actor	Usuario
Precondiciones	Estar en el menú principal Estar conectado al objeto tangible
Postcondiciones	Mostrar la pantalla de Rotar
Flujo	El usuario pulsa el cubo de Rotar y cambia a la pantalla de Rotar

3.13: Caso de uso - Rotar horizontalmente

Descripción	Debe rotar el objeto físico y el objeto en pantalla rotará horizontalmente
Actor	Usuario
Precondiciones	Estar en la pantalla de Rotar Estar conectado al objeto tangible
Postcondiciones	El modelo del objeto tangible rota horizontalmente
Flujo	El usuario rota el objeto físico horizontalmente y el objeto en pantalla lo imita

3.14: Caso de uso - Rotar verticalmente

Descripción	Debe rotar el objeto físico y el objeto en pantalla rotará verticalmente
Actor	Usuario
Precondiciones	Estar en la pantalla de Rotar Estar conectado al objeto tangible
Postcondiciones	El modelo del objeto tangible rota verticalmente
Flujo	El usuario rota el objeto físico verticalmente y el objeto en pantalla lo imita

3.15: Caso de uso - Rotar sobre sí mismo

Descripción	Debe rotar el objeto físico sobre sí mismo y el objeto en pantalla rotará de igual forma
Actor	Usuario
Precondiciones	Estar en la pantalla Rotar Estar conectado al objeto tangible
Postcondiciones	El modelo del objeto tangible rota sobre sí mismo
Flujo	El usuario rota sobre sí mismo el objeto físico y el objeto en pantalla lo imita

3.16: Caso de uso - Salir de la pantalla de Rotar

Descripción	Debe cambiarse la pantalla con el menú principal
Actor	Usuario
Precondiciones	Estar en la pantalla de Rotar
Postcondiciones	Mostrar la pantalla del menú principal
Flujo	El usuario pulsa el botón Back y vuelve a la pantalla del menú principal

3.17: Caso de uso - Entrar en Mover

Descripción	Debe cambiarse a la pantalla de Mover
Actor	Usuario
Precondiciones	Estar en la pantalla del menú principal Estar conectado al objeto tangible
Postcondiciones	Mostrar la pantalla de Mover
Flujo	El usuario pulsa en el cubo de Mover y cambia a la pantalla de Mover

3.18: Caso de uso - Mover lateralmente

Descripción	Debe rotarse el objeto físico hacia un lado un poco y se verá desplazarse el modelo
Actor	Usuario
Precondiciones	Estar en la pantalla de Mover Estar conectado al objeto tangible
Postcondiciones	El modelo del objeto rota un poco y se desplaza hacia uno de los dos lados
Flujo	El usuario rota el objeto físico un poco a izquierda o derecha y el modelo imita la rotación y comienza a desplazarse lateralmente

3.19: Caso de uso - Mover frontalmente

Descripción	Debe rotarse el objeto físico hacia el frente y se verá alejarse el modelo
Actor	Usuario
Precondiciones	Estar en la pantalla de Mover Estar conectado al objeto tangible
Postcondiciones	El modelo del objeto se aleja
Flujo	El usuario rota baja el frente del objeto un poco y el modelo comienza a alejarse

3.20: Caso de uso - Mover posteriormente

Descripción	Debe rotarse el objeto físico hacia atrás y se verá acercarse el modelo
Actor	Usuario
Precondiciones	Estar en la pantalla de Mover Estar conectado al objeto tangible
Postcondiciones	El modelo del objeto se acerca
Flujo	El usuario rota el objeto físico hacia atrás un poco y el modelo comienza a acercarse

3.21: Caso de uso - Mover hacia arriba

Descripción	Debe inclinarse el objeto mucho hacia atrás y se verá como el modelo asciende
Actor	Usuario
Precondiciones	Estar en la pantalla de Mover Estar conectado al objeto tangible
Postcondiciones	El modelo del objeto asciende
Flujo	El usuario inclina el objeto físico mucho hacia atrás y el modelo comienza a ascender

3.22: Caso de uso - Mover hacia abajo

Descripción	Debe inclinarse el objeto mucho hacia adelante y se verá como el modelo desciende
Actor	Usuario
Precondiciones	Estar en la pantalla de Mover Estar conectado al objeto tangible
Postcondiciones	El modelo del objeto desciende
Flujo	El usuario inclina el objeto físico mucho hacia adelante y el modelo comienza a descender

3.23: Caso de uso - Interactuar con objeto

Descripción	Debe moverse el objeto hasta tocar uno de los objetos de la escena y ver un cartel o una acción del objeto
Actor	Usuario
Precondiciones	Estar en la pantalla de Mover
Postcondiciones	Mostrar un cartel o acción del objeto tocado
Flujo	El usuario debe desplazar el modelo con los movimientos del objeto físico hasta tocar un objeto de la escena y ver un cartel o provocar una reacción en el objeto tocado

3.24: Caso de uso - Salir de la pantalla de Mover

Descripción	Debe cambiarse la pantalla a la del menú principal
Actor	Usuario
Precondiciones	Estar en la pantalla de Mover
Postcondiciones	Mostrar la pantalla del menú principal
Flujo	El usuario pulsa el botón Back y vuelve a la pantalla del menú principal

3.25: Caso de uso - Entrar en Libre

Descripción	Debe cambiarse a la pantalla de Mover
Actor	Usuario
Precondiciones	Estar en la pantalla del menú principal Estar conectado al objeto tangible
Postcondiciones	Mostrar la pantalla de Libre
Flujo	El usuario debe pulsar el cubo de Libre y cambia a la pantalla de Libre

3.26: Caso de uso - Desplazarse frontalmente

Descripción	Debe rotarse el objeto físico hacia el frente y el modelo comenzará a moverse hacia el frente sin alejarse
Actor	Usuario
Precondiciones	Estar en la pantalla de Libre Estar conectado al objeto tangible
Postcondiciones	El modelo del objeto tangible se desplaza hacia adelante
Flujo	El usuario rota el objeto físico un poco hacia delante y el modelo se desplaza hacia el frente

3.27: Caso de uso - Desplazarse posteriormente

Descripción	Debe rotarse el objeto físico hacia atrás y el modelo comenzará a moverse hacia atrás sin acercarse
Actor	Usuario
Precondiciones	Estar en la pantalla de Libre Estar conectado al objeto tangible
Postcondiciones	El modelo del objeto tangible se desplaza hacia atrás
Flujo	El usuario rota el objeto físico un poco hacia atrás y el modelo se desplaza hacia atrás

3.28: Caso de uso - Rotar horizontalmente

Descripción	Debe rotarse el objeto físico sobe sí mismo y el modelo girará
Actor	Usuario
Precondiciones	Estar en la pantalla de Libre Estar conectado al objeto tangible
Postcondiciones	El modelo del objeto debe rotar sobre sí mismo
Flujo	El usuario rota el objeto físico y el modelo rotará

3.29: Caso de uso - Mover lateralmente

Descripción	Debe rotarse el objeto físico hacia un lado un poco y se verá desplazarse el modelo
Actor	Usuario
Precondiciones	Estar en la pantalla de Libre Estar conectado al objeto tangible
Postcondiciones	El modelo del objeto rota un poco y se desplaza hacia uno de los dos lados
Flujo	El usuario rota el objeto físico un poco a izquierda o derecha y el modelo imita la rotación y comienza a desplazarse lateralmente

3.30: Caso de uso - Mover hacia arriba

Descripción	Debe inclinarse el objeto mucho hacia atrás y se verá como el modelo asciende
Actor	Usuario
Precondiciones	Estar en la pantalla de Libre Estar conectado al objeto tangible
Postcondiciones	El modelo del objeto asciende
Flujo	El usuario inclina el objeto físico mucho hacia atrás y el modelo comienza a ascender

3.31: Caso de uso - Mover hacia abajo

Descripción	Debe inclinarse el objeto mucho hacia adelante y se verá como el modelo desciende
Actor	Usuario
Precondiciones	Estar en la pantalla de Libre Estar conectado al objeto tangible
Postcondiciones	El modelo del objeto desciende
Flujo	El usuario inclina el objeto físico mucho hacia adelante y el modelo comienza a descender

3.32: Caso de uso - Mover en diagonal

Descripción	Debe inclinarse el objeto hacia un lado y hacia atrás o hacia adelante y el modelo se moverá en diagonal
Actor	ario
Precondiciones	Estar en la pantalla de Libre Estar conectado al objeto tangible
Postcondiciones	El modelo se desplazará diagonalmente
Flujo	El usuario inclina el objeto hacia un lado y hacia atrás o hacia adelante y el modelo del objeto se desplaza diagonalmente

3.33: Caso de uso - Salir de la pantalla de Libre

Descripción	Debe cambiarse la pantalla a la del menú principal
Actor	Usuario
Precondiciones	Estar en la pantalla de Libre
Postcondiciones	Mostrar la pantalla del menú principal
Flujo	El usuario pulsa el botón Back y vuelve a la pantalla del menú principal

3.4.6. Escenarios keypath

En este subapartado se verá como sería la interacción con la interfaz recientemente diseñada que permite el uso del objeto tangible. Se utilizan algunos de los **Casos de uso 3.4.5** para validar el diseño de la interfaz del sistema.

- **Objetivo: entrar en al menú con los cuatro botones (Caso de uso 3.2)**

  Una vez ejecutado el sistema, en la pantalla principal se verá un botón Start que al pulsarlo, llevará al menú con los cuatro botones.

- **Objetivo: entrar en Rotar, Mover, Libre (Caso de uso 3.12, 3.17, 3.25)**

  Desde el menú con los cuatro botones, sólo se tiene que pulsar en uno de los tres que

llevan el mismo texto y se pasa a la escena la cual tiene la representación del objeto tangible.

- **Objetivo: Salir de Rotar, Mover, Libre (Caso de uso 3.16, 3.24, 3.33)**

  En cada una de las secciones siempre hay un botón visible en la esquina inferior izquierda que, al pulsarlo, permite el volver a la escena del menú.

- **Objetivo: Entrar a la sección de la selección de forma (Caso de uso 3.6)**

  Pulsar en el botón de Forma desde el menú.

- **Objetivo: Seleccionar una forma concreta (Caso de uso 3.7, 3.11)**

  Una vez en la escena de modificar, mediante las flechas selectoras laterales, se puede elegir la forma deseada para el objeto tangible. Una vez se tenga la forma que se quiera en el centro de la pantalla (siendo más grande que el resto), hay que pulsar en el botón central con el texto 'Seleccionar' y se volverá al menú. Se muestra el objeto en los botones que antes eran cubos.

- **Objetivo: añadir nuevos objetos y verlos (Caso de uso 3.8)**

  Para añadir una nueva forma, primero es necesario añadirla en la carpeta del programa y luego, dentro de la escena de modificar, pulsar en el botón de 'Actualizar' para que aparezca en las formas visibles.

- **Objetivo: Interactuar con los objetos del escenario (Caso de uso 3.23)**

Dentro de las escenas de Mover, si se mueve la representación del objeto tangible hacia uno de los modelados presentes en la escena y se tocan, este realizará una acción que depende del objeto en sí.

3.4.7. Validar los diseños con los escenarios de validación

En este subapartado se han creado escenarios de validación para confirmar que el sistema cumple con todas las necesidades que pueden tener los usuarios respecto al sistema.

- Pablo quiere entrar en la aplicación

  Si tiene el sistema en ejecución, sólo tiene que pulsar el botón.

- Pablo quiere desplazar el objeto

  Entra en 'Mover' para mover el objeto de forma limitada

  Entra en 'Libre' para utilizar el objeto tangible como un avión

- Pedro quiere rotar el objeto

  Entra en 'Rotar'

- Pedro cambiar la forma del objeto

  Entra en 'Modificar' y selecciona el objeto

- Pedro quiere añadir un nueva forma al objeto porque la aplicación lo tiene

  Si tiene el archivo, solo tiene que copiarlo a la carpeta que utiliza el sistema y darle al botón 'Actualizar' para que se muestre y seleccionarlo.

- Sara quiere hacer interaccionar los objetos

  Entra en 'Mover' o 'Libre' y hace que dos objetos choquen

3.4.8. Flujo

Una vez establecidos los escenarios de validación, se procede a detallar el flujo de la interfaz del sistema utilizando como apoyo los casos de uso detallados en la sección 3.4.5 y la figura 3.11 con las pantallas de la misma.

Para entender de forma más clara el flujo de HITO, se van a dividir en las pantallas que pueden ir consecutivas sin volver a una pantalla anterior.

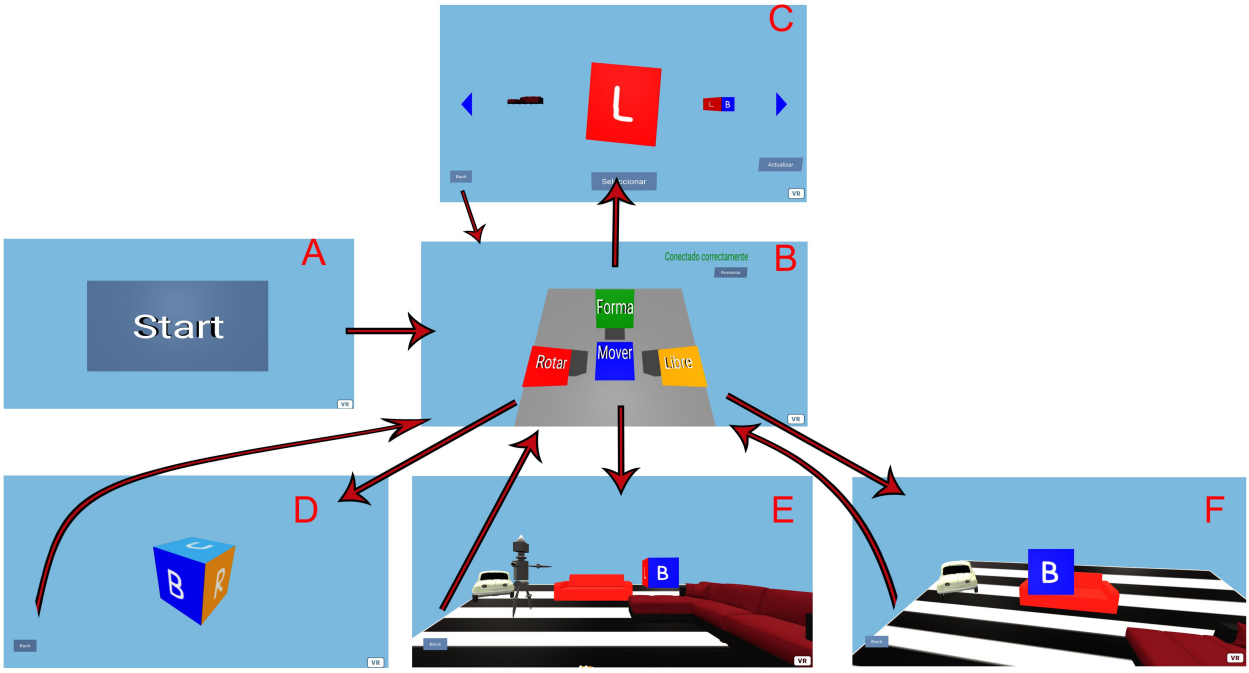


Figura 3.11: Diagrama de flujo

Una vez en ejecución, se muestra una pantalla con sólo un botón grande con 'Start' en él que se debe pulsar, como se dice en caso de uso 3.2, para poder acceder al menú principal como en la figura 3.12. No se puede volver a A.

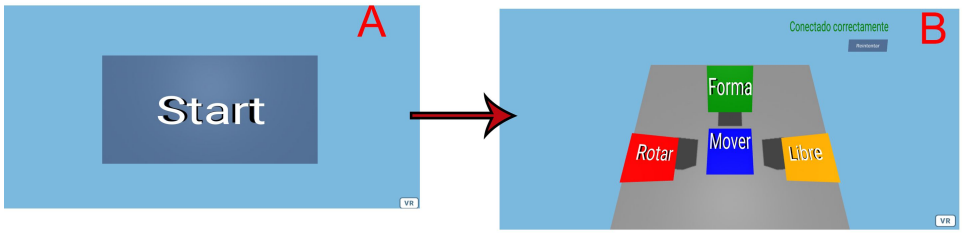


Figura 3.12: Flujo AB

En la pantalla de B se tiene acceso al resto de escenas de la interfaz. Las escenas a las que se pueden acceder desde el menú de B son:

- **Rotar:** una vez pulsado en el cubo 'Rotar' (caso de uso 3.12) lleva a la pantalla D como en la figura 3.13. Se puede volver a la pantalla B como se dice en el caso de uso 3.16.

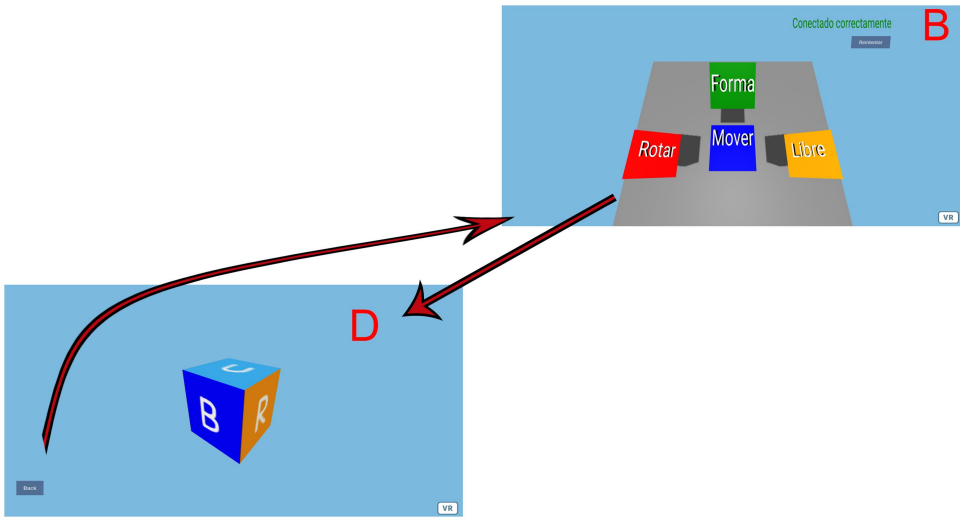


Figura 3.13: Flujo BD y DB

- **Mover:** una vez pulsado en el cubo 'Mover' (caso de uso 3.17) lleva a la pantalla E como en la figura 3.13. Se puede volver a la pantalla B como se dice en el caso de uso 3.24.

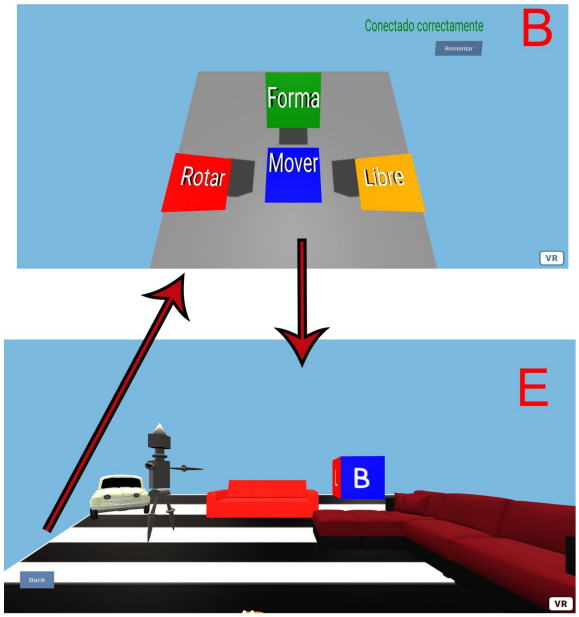


Figura 3.14: Flujo BE y EB

- **Libre:** una vez pulsado en el cubo 'Libre' (caso de uso 3.25) lleva a la pantalla F como en la figura 3.14. Se puede volver a la pantalla B como se dice en el caso de uso 3.33.

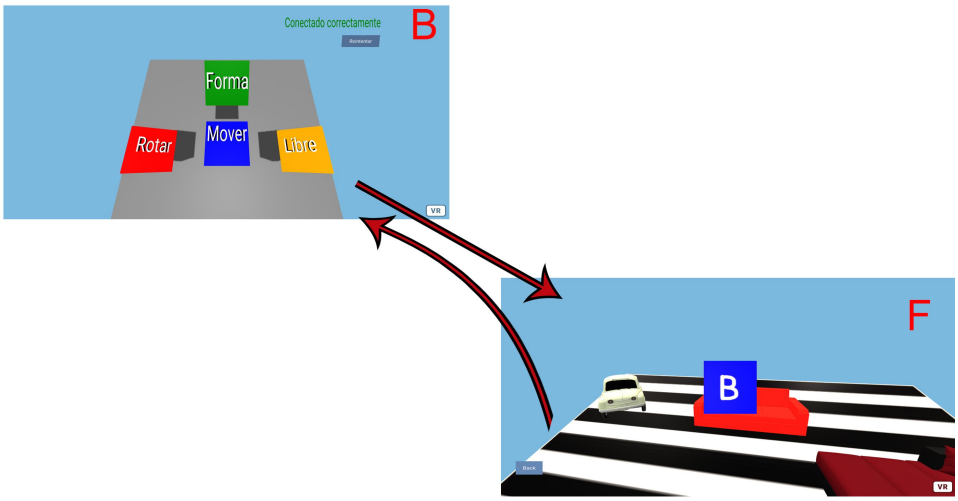


Figura 3.15: Flujo BF y FB

- **Forma:** una vez pulsado en el cubo 'Forma' (3.6) lleva a la pantalla C como en la figura 3.16. Se puede volver a la pantalla B como se dice en el caso de uso 3.10.

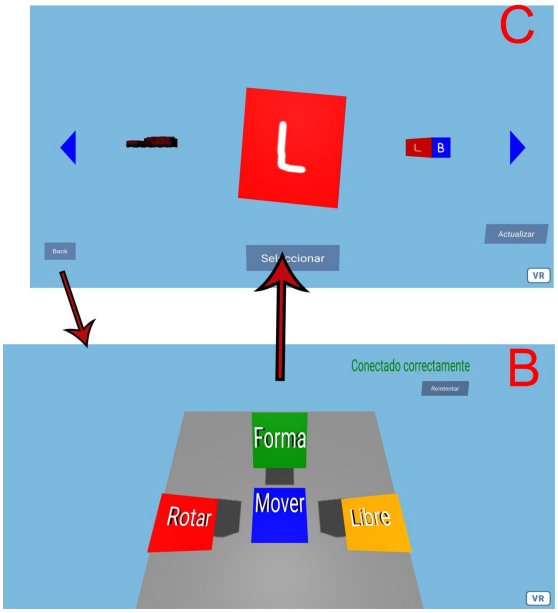


Figura 3.16: Flujo BC y CB

3.5. Conclusiones de los requisitos del sistema

Al finalizar este capítulo, se obtienen los requisitos que HITO debe cumplir para lograr los objetivos iniciales y una descripción del producto centrada en las pantallas obtenidas a través de los prototipos así como la carcasa del objeto tangible. También se obtienen las pantallas a pantallas que tendrá HITO a partir del flujo y los casos de uso que dan la información de lo que puede hacer el usuario. Con la relación entre el objeto físico y el modelo que lo representa y con las pantallas en las que se mueven ambos por el entorno, parece aumentar la sensación de inmersión.

En resumen, se concreta el sistema para permitir avanzar a la fase de desarrollo.

Capítulo 4

Arquitectura del sistema

En este capítulo de la memoria se explicará la estructura, las escenas y los diagramas de actividad y de secuencia del sistema

4.1. Arquitectura

En la figura 4.1, se puede apreciar la configuración base de la arquitectura del sistema en el que consiste HITO. La arquitectura del sistema sigue un modelo de cliente-servidor con dos capas: Front End y Back End. Estas dos capas se organizan una arquitectura de modelo-vista-controlador para gestionar el sistema de forma más cómoda. Este cliente-servidor se encuentra en el ordenador y se conectan entre ellos mediante servicios Rest para ejecutar el cliente a través del navegador. A este conjunto se le conecta un servidor externo, la BeagleBone. El cliente para la Beagle, el ordenador, actúa como un cliente activo, aquél que realiza todas las acciones sobre la conexión, y el servidor pasivo, la Beagle, sólo espera las peticiones del cliente para actuar en consecuencia.

Para clarificar de forma más sencilla el modelo-vista-controlador, en la figura 4.1 se añaden colores a los artefactos. La vista está formada por los componentes en naranja, el controlador es amarillo y el modelo es de color azul.

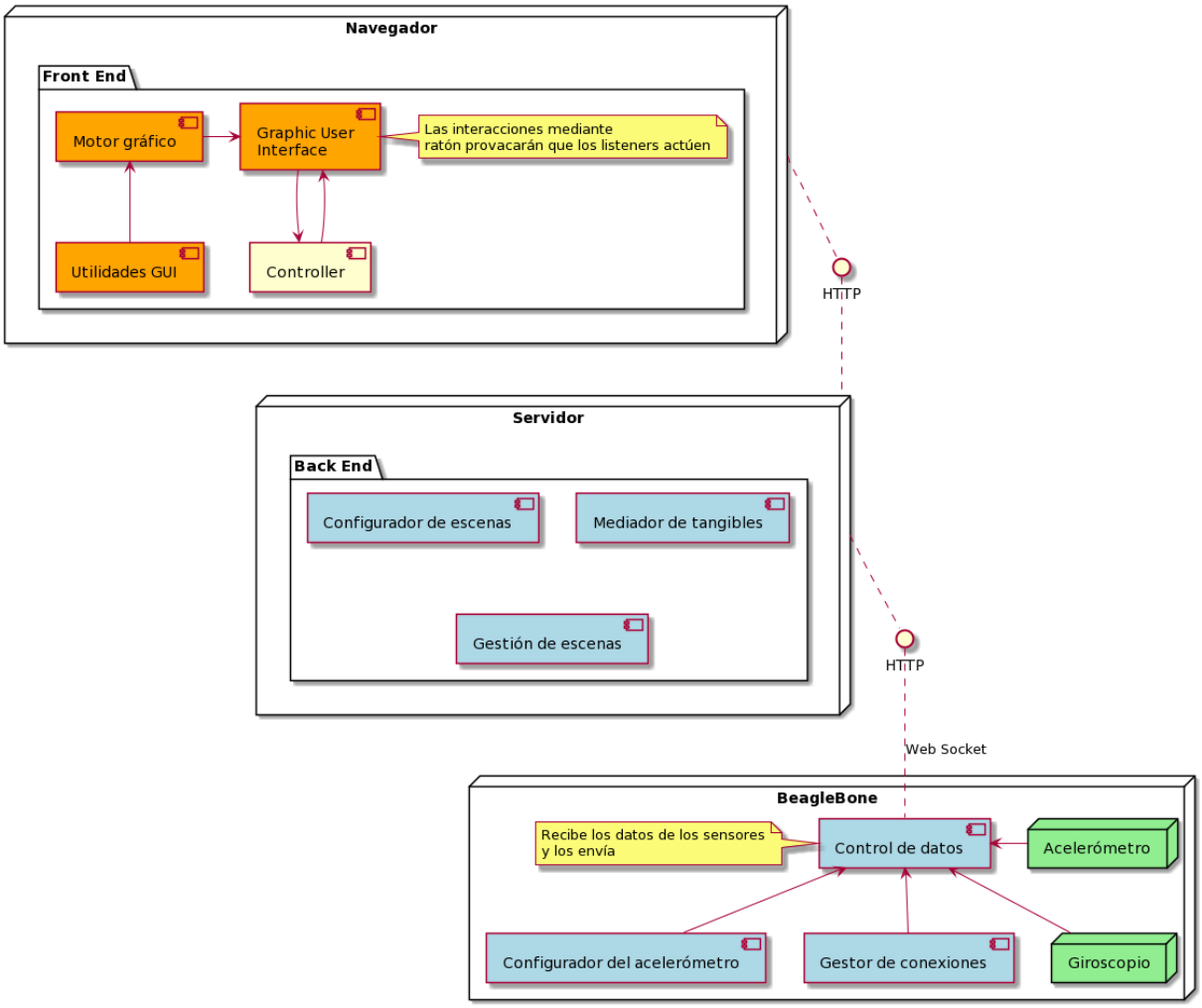


Figura 4.1: Arquitectura del sistema

- **Navegador:** parte del ordenador donde se ejecuta el cliente a través del navegador y contiene el *Front End* que permite el inicio de la interfaz de HITO.

  - **Front End:** contiene la interfaz del usuario y los elementos relacionados con ella. Se encuentran la vista y el controlador del modelo-vista-controlador. Puede conectarse con el navegador gracias a una conexión APIRest que permite las operaciones de GET para obtener información de los directorios del proyecto. Sus componentes internos son:

- **GUI:** hace uso del motor gráfico[1] y de A-Frame para poder mostrar los modelos tridimensionales de la interfaz que se ve en la imagen 3.11. Es la encargada de mostrar la retroalimentación al usuario cuando éste ejerce alguna acción, ya sea con el ratón para transitar en las escenas o con el objeto tangible, la BeagleBone, para ver los movimientos e interacciones de su representación.
  - **Utilidades GUI:** el componente forma parte del motor gráfico y se encarga de añadir nuevas funcionalidades al mismo para que la GUI haga uso de ellos.
- **Listeners:** envía las reacciones de las interacciones al controlador.
- **Controlador:** recibe la información de los listeners y es el encargado de parar el flujo de datos por alguna condición que no se cumple o enviar la información al back end.

- **Servidor:** parte del ordenador que contiene el *Back End* y la lógica interna de HITO. Es el servidor que contiene el Rest que permite la ejecución del sistema. También es el encargado de conectar con el servidor externo, la BeagleBone.
  - **Back End:** contiene parte del modelo de la arquitectura modelo-vista-controlador.
    - **Configurador de escenas:** son todos los métodos que se encargan de mostrar u ocultar y ordenar y posicionar todos los elementos de las escenas que se pueden ver en la imagen 3.11. Es la parte del modelo que permite toda la composición visual de la GUI.
    - **Gestión de escenas:** es la parte de la lógica interna encargada de permitir que las interacciones mediante el ratón funcione. También se encarga de permitir todos los casos de uso 3.4.8 referentes al movimiento del objeto tangible de las escenas. Permite además, conectarse a la BeagleBone para pedir los datos de los sensores que se usan para el movimiento anteriormente

[1] three.js

mencionado. Gestiona en parte junto al *Control de datos* la calibración de los sensores.

- **Mediador de tangibles:** se encarga de generar el servidor Rest para que funcione el sistema en el navegador. Además permite la obtención de los nombres de las formas que puede adoptar el objeto tangible mediante peticiones GET.

- **BeagleBone:** forma parte del modelo y del *Back End*, es el servidor en un componente externo al ordenador y hace de objeto tangible. Está compuesto por el *Control de datos*, que es el que genera el servidor y envía los datos, y los sensores, el *acelerómetro* y el *giroscopio*. La organización de la BeagleBone es la siguiente:

  - **Control de datos:** es el eje central de la BeagleBone, contiene al *Gestor de conexiones* y al *Configurador del acelerómetro*. Gestiona que la información de los sensores se muestre de forma legible para la *Gestión de escena*s. Junto a este último, se encarga de calibrar los sensores para dejarlos a punto cuando se necesiten.

    - **Configurador del acelerómetro:** permite gestionar de forma más sencilla la información del acelerómetro, convirtiendo los valores analógico que da el sensor por valores en digital.

    - **Gestor de conexiones:** es el encargado de lanzar el servidor HTTP que espera a que algún cliente, el *Servidor* del ordenador en cuestión, contacte con él. También se encarga de hacer las llamadas periódicas a la hora de calibrar y de enviar la información de los sensores. Es el que recibe las órdenes cuando se ejecutan los casos de uso 3.3 y 3.4.

  - **Acelerómetro:** sensor ADXL345 [4] que lee la aceleración en 3 ejes. Es afectado por la aceleración de la gravedad.

- **Giroscopio:** sensor ITG3200[23] que lee la velocidad angular en 3 ejes y su temperatura. Es configurable en el ratio de muestreo y su precisión.

En cuanto a las interconexiones del sistema, por una parte se encuentra la BeagleBone, que tiene conectados en sí todos los sensores necesarios mediante conexiones I2C con un Adaptador Hub I2C y se conecta al ordenador por el cable USB. Por otra parte está el ordenador con la interfaz gráfica y la gestión de la misma con los datos recibidos.

4.2. Diseño

En este apartado se va a tratar el diseño del sistema. Para explicar su funcionamiento se van a explicar las escenas y el comportamiento.

4.2.1. Definición de tipos de escenas

En este apartado se explicarán qué significado tiene una escena para el sistema y qué tipos de escenas se consideran durante el desarrollo.

Una escena para HITO es una pantalla interactuable en la que el usuario recibe retroalimentación de alguna forma u otra. Estas escenas se pueden separar dependiendo de la forma de interacción de las mismas. Debido a que hay dos formas en las que el usuario puede interactuar con el sistema, se consideran los siguientes tipos de escenas:

- **Escenas de inmersión:** estas escenas son todas aquéllas escenas en las que el movimiento de los objetos se produzca principalmente a través del movimiento del objeto tangible. Son las escenas objetivo del sistema y donde se demuestra la relación entre objeto tangible y su representación en la pantalla.
- **Escenas de interacción:** estas escenas son todas aquéllas escenas en las que la única forma de interacción sea través del ratón del ordenador. Siempre que se utilice solamente este elemento y no el objeto tangible para pulsar sobre los elementos de

la escena se puede considerar una escena de interacción. Estas escenas tienen como objetivo moverse entre estas y otras escenas o utilizarse para configurar el sistema.

4.2.2. Definición del comportamiento del sistema

En este apartado se van a mostrar los estados por los que pasa el sistema desde que inicia hasta que se cierra con ayuda de los diagramas de actividad 4.2 y secuencia 4.3.

El comportamiento del sistema presenta un ciclo de estados que acaba en un bucle infinito hasta que se cierre alguno de los elementos que interactúan o se tome una decisión que lleve a un punto muerto como se puede ver en la figura 4.2. La secuencia del envío de información entre la *GUI*, la *Gestión de la escena* y la *BeagleBone* se ven en la figura 4.3.

En el estado inicial 'Iniciar interfaz' interviene el *Mediador de tangibles* para poder ejecutar la *GUI* en el navegador. Después de iniciar de forma correcta, la primera pregunta se refiere a si se conecta el Servidor con la Beagle a través de los componentes *Gestión de escenas* y *Control de datos* que aparecen en la figura 4.1. Si no se quiere conectar, la actividad termina en este punto. Si se ha decidido conectarse a la *BeagleBone*, se comprueba que se conecte. En caso negativo, se comprueba si ya se ha intentado la conexión 5 veces, Si no ha pasado eso, se va a la acción *Reintentando conexión*, donde interviene nuevamente la *Gestión de escenas*. Si sí se ha superado ese número, se tendrá que decidir si se quiere reintentar conectarse o no. Si no se hace, la actividad acaba en este punto. Si por el contrario sí se ha decidido volver a intentar, se vuelve a la acción *Reintentado conexión.* Si después de reintentarlo, la conexión es correcta, se empieza la acción de *Calibrar*, donde se ejecutan la calibración de los sensores como se muestra en el cuadro de Inicialización de la figura 4.3, y se cambia a *Esperando*. En esperando se puede decidir enviar dos señales distintas:

- **'end':** para desconectarse, haciendo que que la interfaz envíe a la *Gestión de escenas* la señal de desconexión y que este a su vez le mande a la *BeagleBone* la misma señal que le indicará que el cliente se ha desconectado y que se quede en espera, como se ve

en el cuadro '**end**' de la figura 4.3.

- **'init':** para iniciar la petición de datos que llevará al ciclo de repetición de datos como se puede ver en 'Repetición' de la figura 4.3.

Una vez se realiza la petición por parte del *Gestor de escenas, el Control de datos* le devuelve los datos y el estado pasa a ser *Recibir los datos*. Una vez recibidos, en la actividad de Interpretar los datos, el *Gestor de escenas* interviene para que cuando se pase a *Usar los datos*, la *GUI* pueda representar la información en pantalla. En este punto se puede decidir parar con la señal '**stop**', que llevara al estado Esperando o seguir recibiendo datos al *Realizar peticiones*.

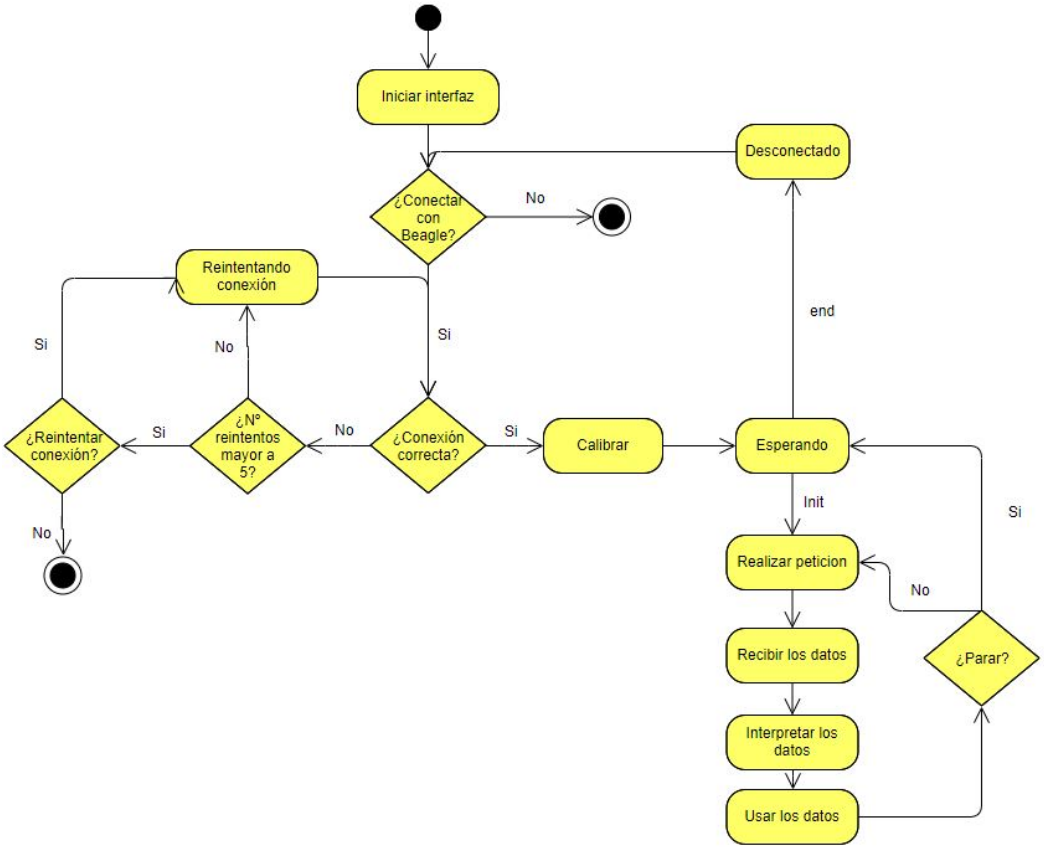


Figura 4.2: Diagrama de actividad

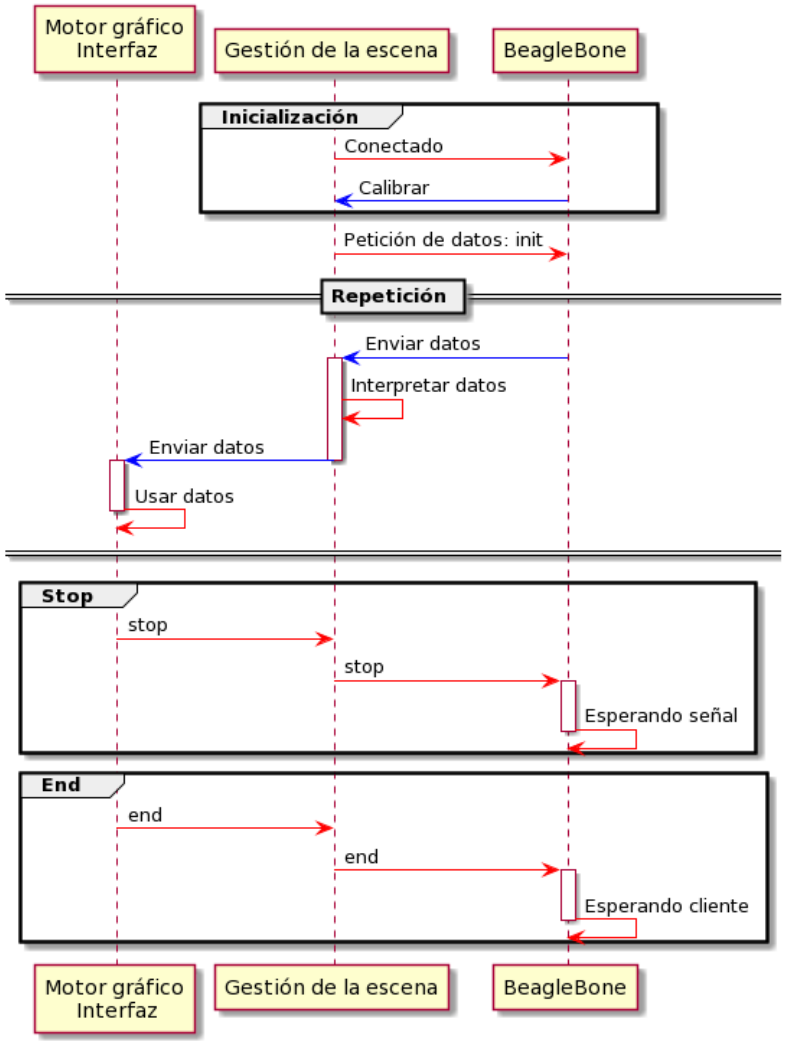


Figura 4.3: Diagrama de secuencia

4.3. Conclusiones de la arquitectura

De los apartados anteriores, se puede obtener una definición clara de la arquitectura, obteniendo sus componentes y su relación entre ellos. También se obtiene que hay dos tipos de escenas, en las que interviene el modelo del objeto tangible y el objeto físico, y en las que estos no intervienen. Además se define el comportamiento base de HITO.

Capítulo 5

Implementación del sistema

En este capítulo se explicará cómo está implementado el código y qué hace cada elemento apoyándose en los componentes expuestos en la arquitectura de la figura 4.1. Para facilitar la explicación, se separará el capítulo en dos grandes secciones, por un lado todo lo que se haya utilizado en el ordenador, por el otro lado, todo lo utilizado en la BeagleBone. El código de la implementación se encuentra en github[1] con una licencia open source GPL compatible con la licencia MIT que tienen a-frame y otros componentes del sistema.

5.1. Nodo Navegador

En este apartado se mostrarán todos los componentes pertenecientes al nodo. Desde esta parte se ejecuta el cliente que conectan con los servicios Rest. La parte principal del navegador compone un modelo-vista-controlador con listeners hecho con JavaScript (30). En la estructura se puede encontrar el front end separado del back end.

5.1.1. Componente FrontEnd

En el front end se encuentra la interfaz y vista del sistema que hace uso de a-frame (49). Todos los objetos que se pueden ver por toda la interfaz son creados directamente en el front end de la aplicación y después desplazados o hecho invisibles por el back end. Está formado por los componentes 5.1.2, 5.1.3 y 5.1.4.

[1] https://github.com/MicTome/tangible—TFM

5.1.2. Componente GUI

Es la interfaz de HITO y está compuesto de los artefactos siguientes:

- **index.html:** carga todos los archivos del sistema en la interfaz, por lo tanto, todos los componentes del front end y back end son cargados desde aquí a excepción del Mediador de tangibles, que carga antes. Carga también los complementos de a-frame. Se encarga de mantener todos los objetos de la 'GUI' a lo largo de las escenas mostradas en la imagen 3.11 y generadas por a-frame, por lo que se ve afectado por el Configurador de escenas y la escena que cargue este. También contiene los listeners que se encarga de recibir las interacciones sobre los objetos de la interfaz para pasarlos al controlador.

  Los objetos de las escenas pueden estar en una posición visible de la escena o tener los mismos invisibles a la vista.

- **A-Frame:** framework de modelador 3D que hace uso de Three.js[2], el motor gráfico, para la creación de dichos modelos. Este framework junto con los dos elementos siguientes, forman parte del componente 'motor gráfico' y se encargan de modelar la interfaz. Sólo puede existir una escena en a-frame, por lo que para crear las .[es]cenas"que concibe el usuario, se desplazan e invisivilizan unos u otros objetos de la escena. En este proyecto se ha usado la versión 1.0.4 explícitamente.

- **A-Frame-physics [13]:** creado explícitamente para a-frame, es un simulador de físicas compuesto por 'aframe-physics-system' y 'aframe-physics-extras' que tienen como objetivo permitir la interacción entre los distintos objetos del proyecto. Forma parte del 'motor gráfico'.

5.1.3. Componente Utilidades GUI

Se realiza en el artefacto registerComponents.js, se registran los distintos componentes nuevos que pueden utilizar los objetos creados a partir de objetos de a-frame. Añade un

[2] three.js

nuevo atributo distinto a los atributos que ya tiene como la rotación, la posición, etc. En esta ocasión se añade el escalado de los modelos obtenidos de los archivos provenientes del blender. Forma parte del 'motor gráfico'.

5.1.4. Componente Controller

Se realiza mediante el artefacto controller.js que contiene todos los métodos que son llamados desde los listeners. No realiza ninguna acción más allá de comprobar si se tiene o no que llamar al método requerido del modelo.

5.2. Nodo Servidor

El servidor Rest se encarga de permitir la ejecución del cliente a través del navegador descrito en la sección anterior. En esta parte se encuentra la lógica interna de HITO programado con JavaScript. El servidor está formado por el back end del sistema y es parte del modelo de la arquitectura modelo-vista-controlador.

5.2.1. Componente BackEnd

Reúne la lógica interna en los tres componentes mostrados en la figura 4.1 de la arquitectura.

5.2.2. Componente Mediador de tangibles

Encargado principalmente de crear la conexión Rest y de permitir la opción GET para acceder a información del directorio del proyecto que pide alguno de los métodos del componente Gestión de escenas.Está compuesto por el siguiente artefacto:

- **tangible.js:** cumple la función de generar la conexión Rest para que el proyecto funcione en el 'Navegador' de la imagen 4.1, es decir, lanza el servidor HTTP para permitir un correcto funcionamiento en los navegadores y que no existan problemas ni con

CORS ni con CSP. A parte de lanzar el sistema, también es el encargado de recibir peticiones GET del cliente.

- **objectsCharge(folder):** dada una ruta, obtiene todos los nombres de los archivos que tenga esa carpeta siempre que terminen en .gltf y .glb que son las extensiones usadas por a-frame. Permite cargar los archivos de los modelos creados en Blender. La ruta por defecto es '/**assets**/**models**/'. La Gestión de escenas es quien pide esta información.

5.2.3. Componente Gestión de escenas

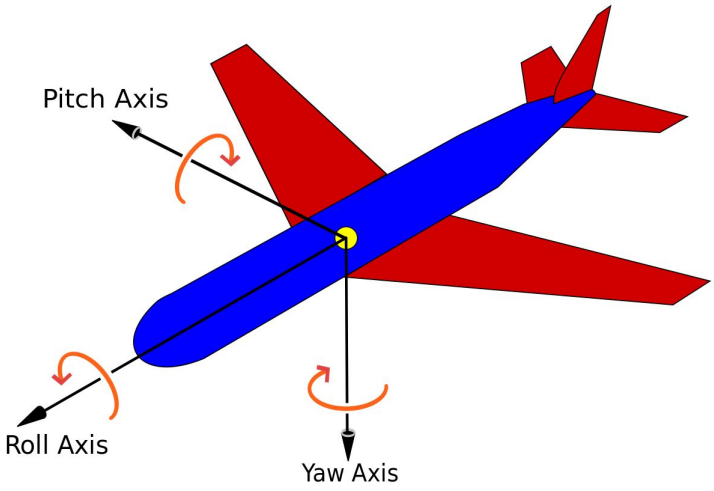
Contiene toda la lógica referente a cálculos y modificaciones de los objetos de las escenas. Se realiza con los siguientes artefactos:

- **methods.js:** eje central del modelo y de la lógica interna. Solo actúa cuando el componente Controller llama a alguno de sus métodos. Necesita la GUI para algunos métodos y hace uso del Configurador de escenas. Los métodos que tiene son:
  - **showText(name):** name es el nombre de un objeto de la escena. Dado un nombre, muestra el objeto con texto asociado al nombre y realiza las acciones pertinentes de cada objeto. Se encarga de dar la retroalimentación al usuario de que un objeto ha sido tocado en la escena Mover. Interviene en la escena de Mover.
  - **interaction(evt):** evt es el evento de interacción del objeto de la interfaz. Dependiendo de con qué objeto esté interactuando el objeto origen, llama al showText o, si no está interactuando con ningún objeto, oculta los objetos mostrados en showText. Interviene en la escena de Mover.
  - **positionObjects(dir):** dir es la dirección a la que se moverá el carrusel. Método usado por la escena de Forma para rellenar el carrusel y desplazarlo a un lado u otro.

- **updateShapeScene():** usado por las escena de Forma que se puede ver para actualizar la lista del carrusel. En la escena existe un botón que añade o elimina los objetos dependiendo de los existentes el directorio. Llama a un método del Configurador de escenas.
- **chargeShapeOfAll():** usado en la escena Forma para, una vez seleccionada una forma, cambiarla tanto en los objetos de lo menú como en los objetos que representan el objeto tangible. Utiliza los atributos que el motor gráfico le da a a-grame para modificar los atributos de los objetos de la interfaz.
- **chargeObjects():** este método realiza una petición HTTP GET al servidor de nodeJS con el objetivo de obtener los nombres de los archivos. Si no existen objetos válidos, se creará un cubo alargado por defecto para la representación del objeto tangible. Necesita el Mediador de tangibles para poder funcionar y se utiliza en la escena de Forma.
- **getSensorData():** obtiene los valores leídos por los sensores, los transforma a enteros para su uso. Para obtener los datos interviene el archivo de websocket.js y el componente Control de datos de la BeagleBone. Se utiliza cuando se hace la calibración inicial en el Menú Principal, pero es llamado cada vez que se piden los datos de los sensores, por lo que actúa en las escenas Rotar, Mover y Libre también.
- **clearSensorVariables():** cuando se sale de las escenas, Rotar, Mover, Free, las variables vuelven a su estado inicial.
- **rotation():** método utilizado por la escena de Rotar y que permite que el modelo del objeto tangible rote en todas las direcciones. Lo primero que se hace es llamar a los métodos getSensorData() y conversiónGyroValues() para tener los valores de los 3 ejes de cada sensor listos para ser utilizados.

  Para poder realizar las rotaciones, hay que calcularlas como si fuesen los movimientos de un avión, teniendo el balanceo o roll (desplazamiento sobre el eje X),

cabezceo o pitch (desplazamiento sobre el eje Y) y la guiñada o yaw (sobre el eje Z).



$$pitch = arctag(\frac{x}{\sqrt{y^2 + z^2}})$$

$$roll = arctag(\frac{y}{\sqrt{x^2 + z^2}})$$

$$yaw = arctag(\frac{\sqrt{x^2 + y^2}}{z})$$

Figura 5.1: Pitch, Roll, Yaw

Estas rotaciones se realizan con los datos del acelerómetro debido a que los valores que devuelve el giroscopio so demasiado variados en poco tiempo y no se pueden considerar 100 % fiables por el ruido que se genera con el mínimo movimiento. Sin embargo, al utilizar los valores del giroscopio, yaw no funciona, porque los giros del acelerómetro en el Z no producen cambio alguno sobre ninguno de los ejes. Para solventar este problema, se utiliza el eje Z del giroscopio.

Para apoyar los valores de un sensor sobre los del otro, se utilizan cuadrantes sobre los 3 ejes para saber en qué posición se encuentra el objeto y qué valores tienen que tomar pitch, roll y el valor de Z del giroscopio. Pero antes hay que tener en cuenta que los valores que el acelerómetro da son valore sobre 180º porque es el rango que permite la diferencia de la aceleración de la Tierra, 9.8 más o menos, con respecto al acelerómetro.

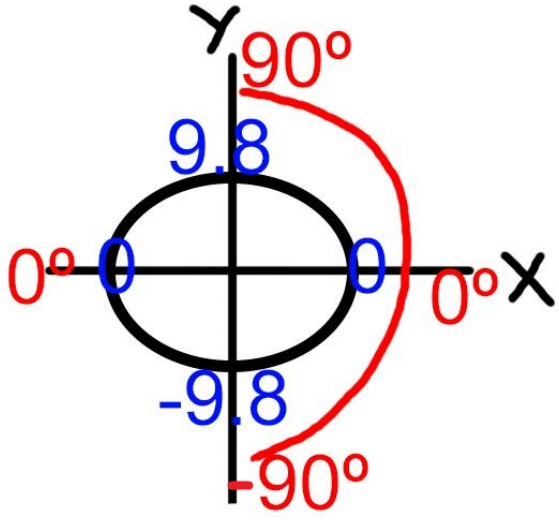


Figura 5.2: Valores de G del acelerómetro

En azul se ven los valores que toma el dato que lee el acelerómetro en sus ejes mientras que en rojo se encuentran los valores en de la conversión a grados. Conociendo esto, cuando el valor que se debería obtener sobrepasa el rango que el acelerómetro puede dar, se le suma el máximo valor que puede tener con el valor actual y así sobrepasar su límite.

Para calcular el valor de pitch y roll, ya que Z se calcula por sí mismo, se necesita saber en qué cuadrante de rotación se encuentra cada eje. Los valores tomados del giroscopio son grados en negativo. Como el pitch se calcula sobre el eje X, los cuadrantes que pueden variar son Y y Z, dando lugar a 4 cuadrantes en los cuales, el valor de pitch puede tomar hasta 3 valores distintos dependiendo de los grados de X del giroscopio como se en las figuras siguientes:

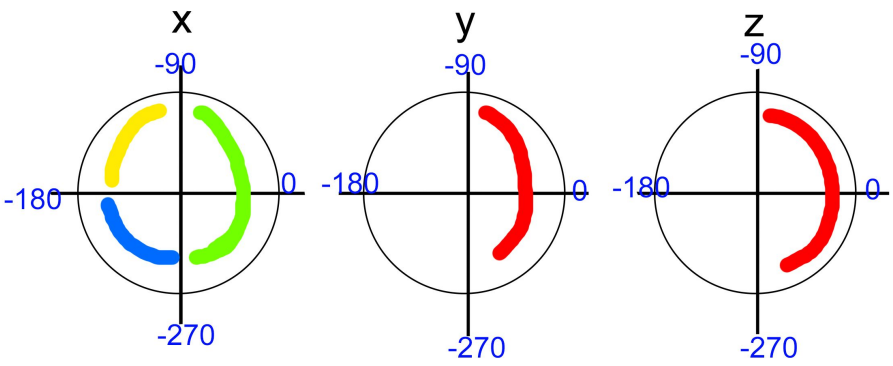


Figura 5.3: Valores de pitch con los cuadrantes Y y Z

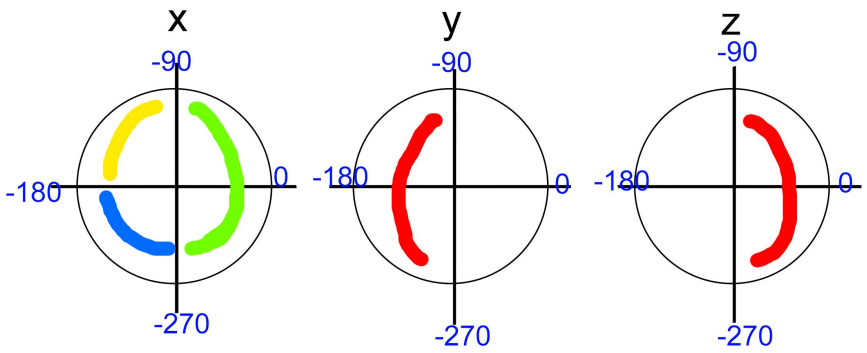


Figura 5.4: Valores de pitch con los cuadrantes -Y y Z

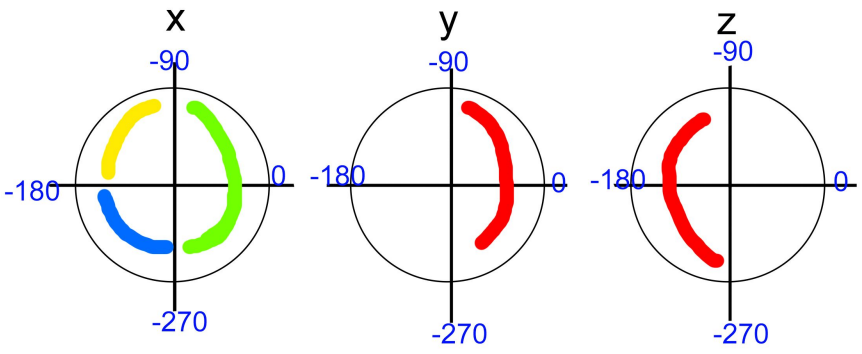


Figura 5.5: Valores de pitch con los cuadrantes Y y -Z

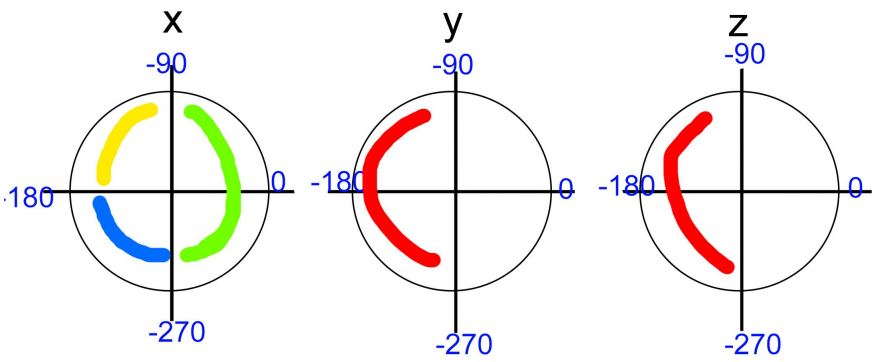


Figura 5.6: Valores de pitch con los cuadrantes -Y y -Z

Para roll pasa exactamente lo mismo. Hay 4 variantes dependiendo de los cuadrantes de X y Z y en cada una de las variaciones, el valor de los grados del eje Y del giroscopio peude dar 3 valores distintos, lo que significa que roll puede tener 3 valores distintos en cada cuadrante. Similar a pitch, las figuras que representan este hecho son las siguientes:

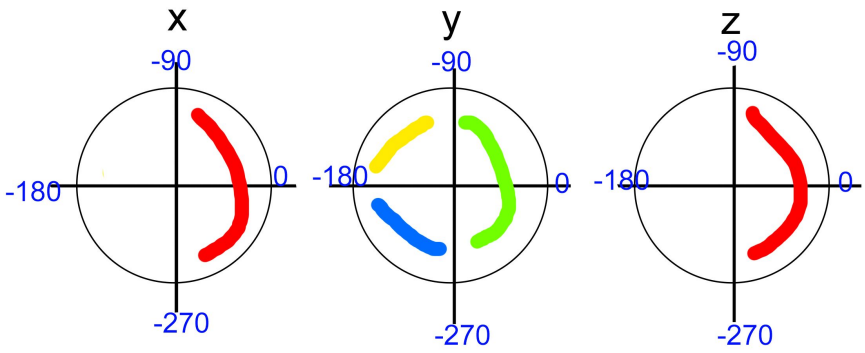


Figura 5.7: Valores de roll con los cuadrantes X y Z

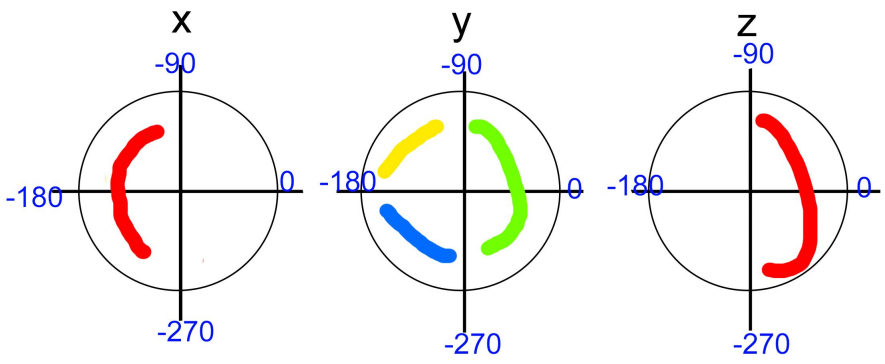


Figura 5.8: Valores de roll con los cuadrantes -X y Z

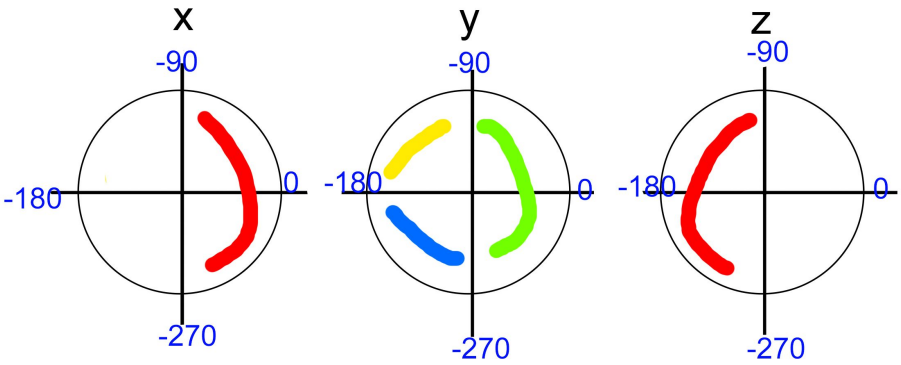


Figura 5.9: Valores de roll con los cuadrantes X y -Z

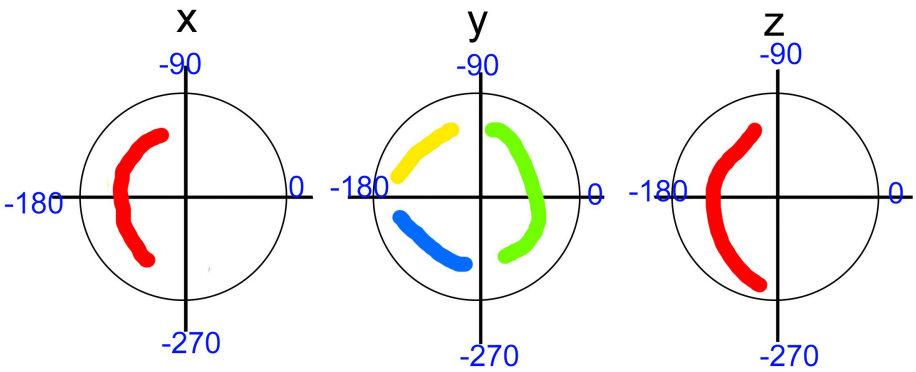


Figura 5.10: Valores de roll con los cuadrantes -X y -Z

- **displace():** método utilizado por la escena de Mover y que permite que el modelo del objeto tangible se desplace en todas direcciones haciendo que la cámara siga

al objeto lateral o verticalmente, no en profundidad, cuando el objeto está cerca de los bordes.

De forma similar a rotation(), en este método se llama a getSensorData() y conversiónGyroValues() para tener los valores de los 3 ejes de cada sensor listos para ser utilizados.

En esta ocasión se divide el método en dos grandes partes, la primera, el movimiento del objeto tangible y la segunda, el movimiento de la cámara.

Para lo primero, se calcula el pitch y el roll como se ha hecho en rotation(), con las fórmulas de la figura 5.1. Con estos cálculos se definen las 4 direcciones a las que puede rotar el objeto tangible y dependiendo de ángulo, la velocidad a la que se desplaza el objeto.

- **Rotar frontalmente:** se utilizan los valores negativos de pitch. Si los grados de pitch se encuentran entre -6º y -80º, entonces se hace el desplazamiento. Si pitch es menor a -16º, el desplazamiento es muy leve; Si está entre -16º y -26º, la distancia recorrida se duplica con respecto al anterior; Si pitch está entre -26º y -40º, entonces el desplazamiento hacia adelante es mucho mayor.

  Para poder desplazar verticalmente el objeto, debido a que no se desplaza verticalmente con el objeto físico, se hace que cuando pitch está entre -55º y -80º, se desplace hacia abajo de forma constante.

  Para la rotación que se muestra del objeto tangible se utiliza el valor 'dif' calculado en la calibracion().
- **Rotar posteriormente:** se utilizan los valores positivos de pitch. Si los grados de pitch se encuentran entre 6º y 80º, entonces se hace el desplazamiento. Si pitch es menor a 16º, el desplazamiento es muy leve; Si está entre 16º y

26º, la distancia recorrida se duplica con respecto al anterior; Si pitch está entre 26º y 40º, entonces el desplazamiento hacia atrás es mucho mayor.

Para poder desplazar verticalmente el objeto, debido a que no se desplaza verticalmente con el objeto físico, se hace que cuando pitch está entre 55º y 80º, se desplace hacia arriba de forma constante.

Para la rotación que se muestra del objeto tangible se utiliza el valor 'dif' calculado en la calibracion().

- **Rotar hacia la izquierda:** se utilizan los valores negativos de roll. Si los grados de roll se encuentran entre -6º y -65º, entonces se hace el desplazamiento. Si roll es menor a -16º, el desplazamiento es muy leve; Si está entre -16º y -26º, la distancia recorrida se duplica con respecto al anterior; Si roll está entre -26º y -65º, entonces el desplazamiento hacia la izquierda es mucho mayor.

  Para la rotación que se muestra del objeto tangible se utiliza el valor 'dif' calculado en la calibracion().
- **Rotar hacia la derecha:** se utilizan los valores negativos de roll. Si los grados de roll se encuentran entre 6º y 65º, entonces se hace el desplazamiento. Si roll es menor a 16º, el desplazamiento es muy leve; Si está entre 16º y 26º, la distancia recorrida se duplica con respecto al anterior; Si roll está entre 26º y 65º, entonces el desplazamiento hacia la derecha es mucho mayor.

  Para la rotación que se muestra del objeto tangible se utiliza el valor 'dif' calculado en la calibracion().

Para el segundo, el movimiento de la cámara, lo que se hace es calcular cada vez que el objeto tangible se mueve, la distancia entre ésta y la cámara. Si la distancia supera un rango determinado, comenzará a seguir al objeto en cualquiera de las

4 direcciones, es decir, izquierda, derecha, arriba y abajo, no lo seguirá si se aleja o se acerca.

- **free():** método utilizado por la escena de Libre y que transforma el movimiento del objeto tangible en una simulación de avión, moviéndose la cámara junto al objeto tangible.

  De forma similar a los métodos anteriores, en este método se llama a getSensorData() y conversiónGyroValues() para tener los valores de los 3 ejes de cada sensor listos para ser utilizados. Sin embargo, este utiliza una fusión entre los dos métodos anteriores, porque utiliza parte de los cuadrantes de rotation() y la misma fórmula de rango de grados que displace().

  Se utilizan los cuadrantes de las figuras 5.3, 5.4 y 5.5 para calcular si pitch su valor positivo o es -pitch, su valor negativo. PAra roll se calcula de la misma forma, utilizando el cuadrante de las figuras 5.7 y 5.9 para comprobar si roll es positivo o es su valor en negativo.

  Una vez hecho esto, se procede a utilizar la versión de displace() utilizando rango de grados, pero dependiendo, esta vez, de los valores que pueda tomar el eje Z del giroscopio. Con esos valores, se tiene que por cada rango de valores de Z se obtienen los 4 movimientos:

  - **Rotar frontalmente:** se utilizan los valores negativos de pitch. Si los grados de pitch se encuentran entre -6º y -80º, entonces se hace el desplazamiento. Si pitch es menor a -16º, el desplazamiento es muy leve; Si está entre -16º y -26º, la distancia recorrida se duplica con respecto al anterior; Si pitch está entre -26º y -40º, entonces el desplazamiento hacia adelante es mucho mayor.

    Para poder desplazar verticalmente el objeto, debido a que no se desplaza

verticalmente con el objeto físico, se hace que cuando pitch está entre -55º y -80º, se desplace hacia abajo de forma constante.

- **Rotar posteriormente:** se utilizan los valores positivos de pitch. Si los grados de pitch se encuentran entre 6º y 80º, entonces se hace el desplazamiento. Si pitch es menor a 16º, el desplazamiento es muy leve; Si está entre 16º y 26º, la distancia recorrida se duplica con respecto al anterior; Si pitch está entre 26º y 40º, entonces el desplazamiento hacia atrás es mucho mayor.

  Para poder desplazar verticalmente el objeto, debido a que no se desplaza verticalmente con el objeto físico, se hace que cuando pitch está entre 55º y 80º, se desplace hacia arriba de forma constante.

- **Rotar hacia la izquierda:** se utilizan los valores negativos de roll. Si los grados de roll se encuentran entre -6º y -65º, entonces se hace el desplazamiento. Si roll es menor a -16º, el desplazamiento es muy leve; Si está entre -16º y -26º, la distancia recorrida se duplica con respecto al anterior; Si roll está entre -26º y -65º, entonces el desplazamiento hacia la izquierda es mucho mayor.

- **Rotar hacia la derecha:** se utilizan los valores negativos de roll. Si los grados de roll se encuentran entre 6º y 65º, entonces se hace el desplazamiento. Si roll es menor a 16º, el desplazamiento es muy leve; Si está entre 16º y 26º, la distancia recorrida se duplica con respecto al anterior; Si roll está entre 26º y 65º, entonces el desplazamiento hacia la derecha es mucho mayor.

Sin embargo, en lugar de añadir estos desplazamientos como en displace(), se utiliza la siguiente fórmula:

$$objectPositionX = objectPositionX +/- displacementValue x sin(z +/- rollValue)$$

$$objectPositionZ = objectPositionZ +/- displacementValue x cos(z +/- rollValue)$$

Donde dependiendo de la posición al a que se mueva el objeto, se suman o se restan los valores. *displacementValue* es el valor escogido para el movimiento del objeto en el espacio tridimensional. z es el valor del eje z del giroscopio en radianes. *rollValue* es un valor de corrección del ángulo para que el desplazamiento funcione adecuadamente.

En esta ocasión, la escena de Libre tiene una cámara pegada al objeto tangible y por lo tanto, cuando el objeto se mueve, la cámara le sigue sin hacer cálculos.

- **conversionGyroValues():** convierte los valores del giroscopio en valores más aptos para su uso, es decir, ajusta los valores para que se encuentren entre 0 y -360 grados. Es utilizado en las escenas de Rotar, Mover y Libre. Se utiliza el valor del ruido 'dif' de los ejes del giroscopio calculado en calibration(). Para ajustar estos valores se hace lo siguiente por cada eje:

$$Si\ ((ValorAnterior - ValorActual) > dif/2)\ entonces:$$

$$Si\ (ValorActual > ValorRuidoPos)\ entonces:$$

$$Grados = ValorActual/180$$

$$RuidoAcumuladoPos++$$

$$Si\ (ValorActual < ValorRuidoNeg)\ entonces:$$

$$Grados = ValorActual/180$$

$$RuidoAcumuladoNeg++$$

Con esto se calculan los grados de todos los ejes y la cantidad de veces que se acumula el ruido cuando el valor es actual es mayor o menor que el valor del ruido. Para eliminar el ruido, se hace lo siguiente:

$$Si\ (RuidoAcumuladoPos >= 8)\ entonces:$$

$$Grados- = 0{,}756$$

$$Si\ (RuidoAcumuladoNegs >= 8)\ entonces:$$

$$Grados+ = 0{,}250$$

Siendo los valores sumados y restado unos valores encontrados a medida que se hacían experimentaciones sobre el ruido del sensor. Para terminar, se transforma el valor en radianes que es el valor que usan los objetos de a-frame para sus rotaciones:

$$radianes = Grados * PI/180:$$

- **calibration():** método utilizado para encontrar los valores de error de los ejes de ambos sensores. Al ser ejecutado automáticamente corresponde a la escena del Menú Principal. Este método es llamado por el Control de datos de la Beagle para realizar una prueba de 30 muestras. Para calibrar el ruido que tienen los sensores se utiliza la siguiente fórmula sobre los valores de los ejes de ambos sensores:

$$Si\ \mathbf{count} < 30\ entonces:$$

$$Si\ \mathbf{ST} < minST\ entonces:$$

$$minST\ = \mathbf{ST}$$

$$Si\ \mathbf{ST} > maxST\ entonces:$$

$$maxST\ = \mathbf{ST}$$

$$\mathbf{difST} = maxST - minST$$

Donde **count** es el número de la muestra; **S** es la letra que indica el acelerador y el giroscopio; **T** toma los valores x, y y z de los ejes; **minST** y **maxST** son los

valores mínimos y máximos que toma el valor durante y **difST** es el ruido que tienen cada sensor en cada uno de sus ejes.

De este algoritmo salen 6 valores **dif** que son los que muestran el ruido y que serán utilizados en los métodos de las escenas Rotar, Mover y Libre.

- **websocket.js:** está separado de methods.js para una mejor organización. Se encarga principalmente de establecer la conexión en forma de cliente con el servidor de la BeagleBone. Establece la conexión automáticamente al momento de entrar en la escena del Menú Principal. Tiene métodos para lanzar la señal de iniciar, parar o finalizar la conexión. Los dos primeros ocurren cuando se entra y sale de las escenas Rotar, Mover y Libre. Mientras que el último solo puede ocurrir en el Menú Principal. Iniciar hace que el servidor envíe la información, parar hace que el servidor deje de enviarla y finalizar corta la conexión del cliente. El servidor seguirá activo y esperando clientes cuando la conexión es cortada. Si no se consigue conectar al servidor, reintenta 5 veces la conexión y al no poder conectarse, envía la información a la interfaz para que cambie en consecuencia. Tiene controladas las conexiones y desconexiones del servidor, permitiendo saber el estado de la conexión. El dato que es recibido desde el servidor es un JSON que se tiene que tratar para obtener los valores de los sensores. Está ampliamente relacionado con el Control de datos para las conexiones con el servidor.

5.2.4. Componente Configurador de escenas

Es el encargado de mover los objetos de la interfaz de un lado a otro cuando las escenas lo necesitan. La única escena que no lo necesita es la escena Inicial, puesto que es la que ya está precargada. Contiene un solo archivo donde se encuentran todos los métodos de configuración. Las escenas que intervienen con este componente son las descritas en 4.2.1. Este componente esta formado por el siguiente artefacto:

- **configSceneMethods.js:**

  - **chargeSelection():** método que mueve los objetos de la escena anterior, en este caso Inicio, para colocar y hacer visibles los objetos del Menú Principal. La carga de la escena tiene un retardo de 1 segundo mientras las animaciones de los elementos se ejecutan.
  - **chargeRotation():** mueve los objetos del Menú Principal a un lugar fuera del rango de visión y coloca los objetos de la escena Rotar. Se realiza una animación antes de cambiar de escena.
  - **chargeFromRotation():** pasa de la escena Rotar al Menú Principal. Al cambiar de escena, los elementos del Menú Principal hacen una animación de colocarse en su sitio.
  - **chargeDisplace():** mueve los objetos del Menú Principal con una animación fuera del campo de visión y coloca los objetos de la escena Mover.
  - **chargeFromDisplace():** mueve los objetos de la escena Mover fuera del campo de visión y coloca los objeto del Menú Principal con una animación.
  - **chargeFree():** mueve los objetos del Menú Principal con una animación fuera del rango de visión y coloca los objetos de Libre.
  - **chargeFromFree():** mueve los objetos de Libre fuera del campo de visión y coloca los objetos del Menú Principal con una animación.
  - **chargeShape():** mueve los objetos del Menú Principal con una animación fuera del rango de visión y coloca los objetos de Forma.
  - **chargeFromShape():** mueve los objetos de Forma fuera del campo de visión y coloca los objetos del Menú Principal con una animación.

5.3. Nodo BeagleBone

La placa necesita conectarse con su cable USB para recibir corriente. En este caso se conecta directamente al ordenador que haga uso del proyecto para permitir una conexión sin internet. En la conexión cliente-servidor, donde el cliente es Navegador+Servidor, es el servidor externo y por lo tanto es el objeto tangible en sí. Utiliza Python para generar tanto la conexión por websocket con HTTP como la manipulación de los datos de los sensores.

5.3.1. Componente Control de datos

Formado por un único archivo que contiene la relación con los componentes de Configurador del acelerómetro (Adafruit), Gestor de conexiones (Tornado) y los sensores, el acelerómetro y el giroscopio. Compuesto por el artefacto:

- **server.py:** Es el archivo donde se encuentra toda la lógica de conexión con el cliente y de leer la información de los sensores.
    - **ITG3200:** contiene los métodos y configuraciones necesarias para obtener datos del giroscopio.
        - **(__init__:** inicializa todos los valores del sensor, sus variables y los ejes mediante direcciones en hexadecimal. El ITG3200 se detecta en la dirección 0x68.
        - **init:** inicializa el bus de direcciones con los valores adecuados. En esta ocasión el ratio de lectura de información es de 90ms, con 2000º/s en paso bajo, se piden obtener todos las mediciones y se establece el reloj del giroscopio para facilitar el correcto funcionamiento.
        - **conversion:** método que extrae la información del bus del giroscopio con la lectura de los ejes y la transforma en datos legibles. La formula utilizada es:

$$\mathbf{nGyro} = data[i] * 256 + data[i+1]$$

$$Si\ \mathbf{nGyro}\ > 32767\ entonces:$$

$$\mathbf{nGyro} -= 65536$$

n: x, y y z
i: valores de 0 a 5
65536: el valor de sensibilidad del sensor en todos los ejes
32767: la mitad del valor de sensibilidad.
data[i]: valor más alto registrado por el eje.
data[i+1]: valor más bajo registrado por el eje.

Figura 5.11: Fórmula del giroscopio

La fórmula de 5.11 transforma del valor analógico en 16 bits complemento a 2 del sensor a un valor digital porque se necesitan valores discretos para las mediciones, es decir, valores que no estén continuamente cambiando mientras se hace la medición. Para ello se necesitan el valor más alto registrado por el sensor 'data[i]' y el menor 'data[i+1]' dado que los valores de cada eje vienen dados en estos dos. El más alto hay que desplazarlo 8 bits para evitar problemas de overflow y contener el resultado en 2 bytes. El desplazamiento es 2^8 o lo que es lo mismo, multiplicar por 256. Luego se le suma el segundo byte y se obtiene el valor adecuado. Si este valor supera la mitad del rango de sensibilidad del sensor se le resta la sensibilidad completa para reducirlo al rango de valores que el sensor registra y no de valores de una vuelta completa más otro valor. Con esta conversión de los datos, se obtienen los grados por segundo con los que trabajará el programa en el otro lado del proyecto.

- **Aplication(tornado):** esta clase y la siguiente son las encargadas de generar el servidor que genera el componente 'Tornado'. Es la clase que se utiliza para

inicializar el framework Tornado que permite tanto crear el websocket con el protocolo HTTP y crear bucles de ejecución.

- **MainHandler(tornado):** en esta clase, haciendo uso de tornado, es donde se reciben las peticiones del cliente y desde donde se envía la información recogida por los sensores.
  - **checkorigin(origin):** comprueba que el origen es el buscado. Como no se hace uso de internet, el valor está siempre a True.
  - **open():** listener que reacciona cuando un cliente se conecta al servidor.
  - **onclose():** reacciona cuando el cliente se desconecta.
  - **onmessage(message):** message es la información que envía el cliente para indicar que la conexión ha sido abierta, momento en el que se hace la calibración haciendo uso de ioloop de tornado, que se inicia la transmisión de datos, que se para o que se finaliza. Este método es el encargado de gestionar las señales que envía 'methods' y de permitir o no el envío de la información.
  - **sensors():** es el método que envía la información de los sensores al cliente. Para ello, primero lee los datos y luego los transforma en un archivo JSON que el cliente tiene que recibir y transformar.
  - **sensorsCalibration():** envía los datos de calibración en un archivo JSON.

- **main():** método principal donde se inicializan los atributos necesarios para que tornado ejecute el servidor HTTP en una dirección y puerto concreto y se inicializa el servidor escuchando en esa dirección y puerto.

Capítulo 6

Experimentación

Teniendo HITO definido e implementado, se dispone a realizar una experimentación para comprobar que hay sensación de inmersión y de profundidad mediante la correlación del movimiento del objeto tangible junto con su representación en las distintas escenas.

Con los tipos de escenas obtenidos en la sección 4.2.1 se definen las escenas concretas que se van ha utilizar para realizar el seguimiento de la experimentación. Las escenas que se consideran son las siguientes:

- **Escenas de interacción:**
  - **Inicio:** escena inicial del sistema.
  - **Menú Principal:** escena que contiene los elementos para acceder a las otras escenas excepto la de inicio.
  - **Forma:** escena que sirve para el cambio de forma del modelo del objeto tangible.
- **Escenas de inmersión:**
  - **Rotar:** escena en la que el modelo del objeto tangible rota con el objeto tangible, el objeto físico, de forma simultánea. El propósito de esta escena es crear una relación entre el objeto físico que se está manipulando con la copia modelada.

- **Mover:** escena en la que el modelo del objeto tangible se desplaza por la escena cuando el objeto tangible rota con cierta inclinación y la cámara no le sigue completamente. Hay objetos que son interactuables y reaccionan cuando el modelo del objeto tangible les toca. Busca crear la sensación de profundidad e inmersión al poder mover el modelo del objeto tangible y poder interactuar con otros objetos.
- **Libre:** escena en la que el modelo del objeto tangible se desplaza y rota por la escena como si se tratase de un avión cuando el objeto tangible rota con cierta inclinación. Con una libertad de movimiento casi completa, busca aumentar la sensación de inmersión y profundidad.

En la experimentación se detallará cómo está formada cada escena.

El seguimiento de la experimentación se realizará ordenándolo por las escenas anteriormente expuestas y utilizando los casos de usos expuestos en la sección 3.4.5 como base. Un pequeño vídeo demostrativo se encuentra en en youtube[1].

En cada experimentación que requiera el uso de datos de la BeagleBone, se expondrá el retraso que existe entre esta y Ubuntu mediante una media de la diferencia de los valores dados por ambos en 30 muestras dadas al momento de calibrar los sensores. Esta información se encuentra en tablas en el anexo E.

Como el objeto tangible y su modelo en pantalla han de moverse a la vez, para crear la correspondencia entre objeto físico y modelo, se utilizan tomas de tiempo en las que se realiza una acción concreta comparando los resultados con las series esperadas. En todas las gráficas se han utilizado siempre valores positivos para facilitar las lecturas de las mismas.

[1] https://youtu.be/d1XbBW_qCnA

6.1. Inicio

Primera escena que se encuentra el usuario al iniciar HITO. Consta simplemente del botón de Start para entrar en el menú principal pulsando sobre él. No se puede hacer nada más en esta escena y llevará a la escena 6.2. Esta escena consta de las siguientes experimentaciones basadas en los casos de uso:

1) Entrar en el menú

Basado en el caso de uso 3.2, el primer paso que hará el usuario es entrar en el menú principal desde la escena de inicio. Lo primero que se ve es la pantalla *1* de la figura 6.1. Cuando el usuario coloca el ratón sobre el botón Start, este se agranda como se ve en la pantalla *2* y, al pulsarlo se pasa al menú principal viendo una animación de cómo se van colocando los cubos y toda la interfaz del menú pasando por las imágenes *3*, *4* y *5*.

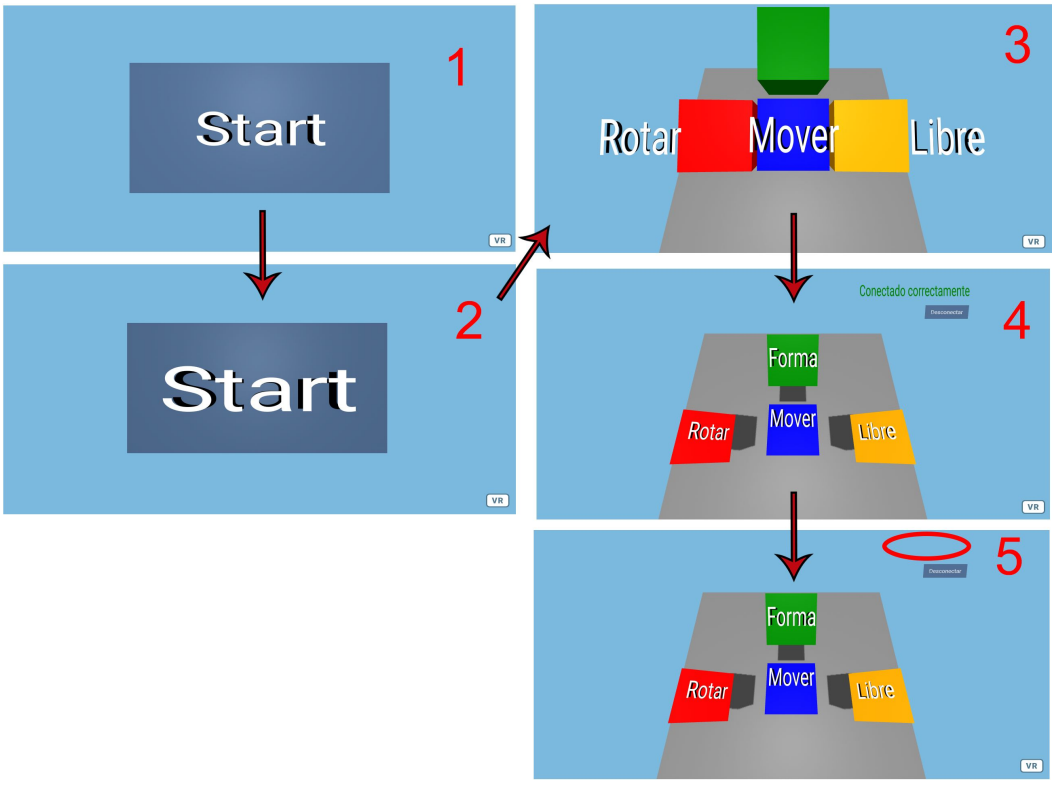


Figura 6.1: Entrar en el menú paso a paso

6.2. Menú principal

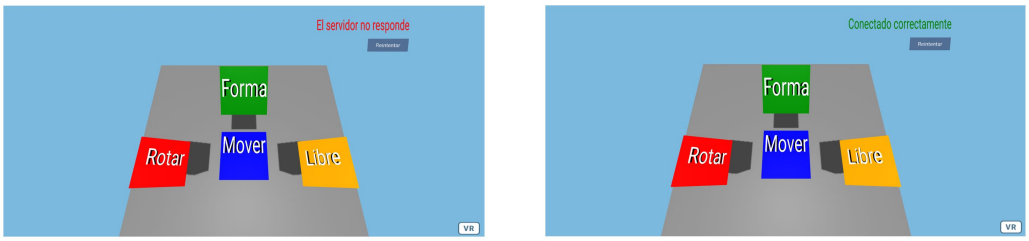


Figura 6.2: Escena de Menú Principal: izquierda desconectado y derecha conectado

Menú principal de HITO. Cuando se entra en esta escena, el menú se coloca mediante una animación y a la vez, se prueba a conectarse con la Beagle. En la figura 6.2 se muestra a la izquierda si no se ha podido conectar o a la derecha si lo ha hecho con éxito. Los elementos interactuables harán una animación cuando el ratón se ponga por encima, mostrando que son interactuables. A continuación se describen los elementos que se encuentran en la escena y su funcionalidad:

- **Plano:** es el plano del fondo que se encuentra debajo de los cubos, no es interactuable y solo se mueve con las animaciones de la escena.
- **Forma:** al pulsar sobre el cubo verde con el texto de Forma, se hará una animación hasta que este ocupe la mayor parte de la pantalla y cambie a la escena 6.3. Se puede entrar tanto conectado como desconectado de la Beagle.
- **Rotar:** al pulsar sobre el cubo rojo con el texto de Rotar, se hará una animación hasta que este ocupe la mayor parte de la pantalla y cambie a la escena 6.4. Solo se puede entrar si se está conectado.
- **Mover:** al pulsar sobre el cubo azul con el texto de Mover, se hará una animación hasta que este ocupe la mayor parte de la pantalla y cambie a la escena 6.5 Solo se puede entrar si se está conectado.

- **Libre:** al pulsar sobre el cubo amarillo con el texto Libre, se hará una animación hasta que este ocupe la mayor parte de la pantalla y cambie a la escena 6.6 Solo se puede entrar si se está conectado.

- **Desconectar:** aparecerá el botón arriba a la derecha solo si se está conectado a la BeagleBone. Poner el ratón sobre el botón hará que resalte. Si se pulsa, aparecerá un texto encima y en rojo en el que pondrá *'Desconectado'* y se habrá desconectado del servidor de la Beagle.

- **Reintentar:** aparecerá el botón arriba a la derecha si se ha desconectado o si no se ha podido conectar a la BeagleBone. Poner el ratón sobre el botón hará que resalte. Si se pulsa, aparecerá un texto encima y en verde en el que pondrá *'Conectando...'* y si falla, uno en rojo en el que podrá *'Reintantando: X fallos'* donde 'X' mínimo 1 y máximo 5. Si en alguno de los intentos se conecta, el texto se podrá en verde y pondrá *'Conectado correctamente'* y si ha fallado las 5 veces, podrá *'El servidor no responde'*.

Esta escena consta de las siguientes experimentaciones basadas en los casos de uso:

1) Desconectar del objeto tangible

Basado en el caso de uso 3.3, en el menú principal como en la pantalla *1* de la figura 6.3 se puede ver arriba a la derecha un botón con el texto *Desconectar*. Si el usuario mueve el ratón hasta el botón, se hará más grande y cambiará su color como en la pantalla *2* y, al pulsarlo, se desconectara de la BeagleBone y se producirá un cambio en la interfaz como se puede ver en la pantalla *3*, el botón pasa a tener el texto *Reintentar* y un texto *Desconectado* en rojo.

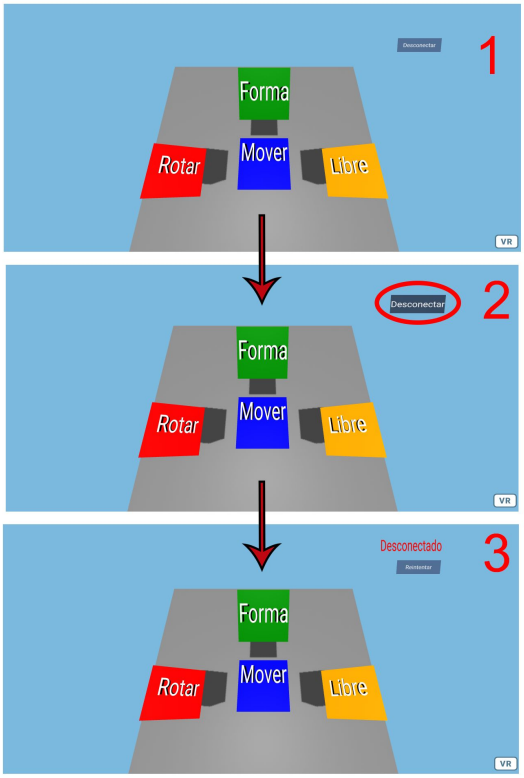


Figura 6.3: Desconectar del objeto tangible paso a paso

2) Conectar al objeto tangible

Basado en el caso de uso 3.4. Cuando el usuario se encuentra en el menú principal y arriba a la derecha le aparece el botón Reintentar y el texto *'Desconectado'* o *'El servidor no responde'*, puede intentar conectarse al objeto tangible.

Al colocar el cursor del ratón sobre el botón *Reintentar*, éste se hará más grande como en la pantalla *2* y al pulsarlo intentará conectarse como se muestra en las imágenes *3* y *4* de las figuras 6.4 y 6.5. El resultado que se espera que verá el usuario pueden ser dos:

- Conectarse como en la pantalla *5* de la figura 6.4
- Fallar el intento como en la pantalla *5* de la figura 6.5

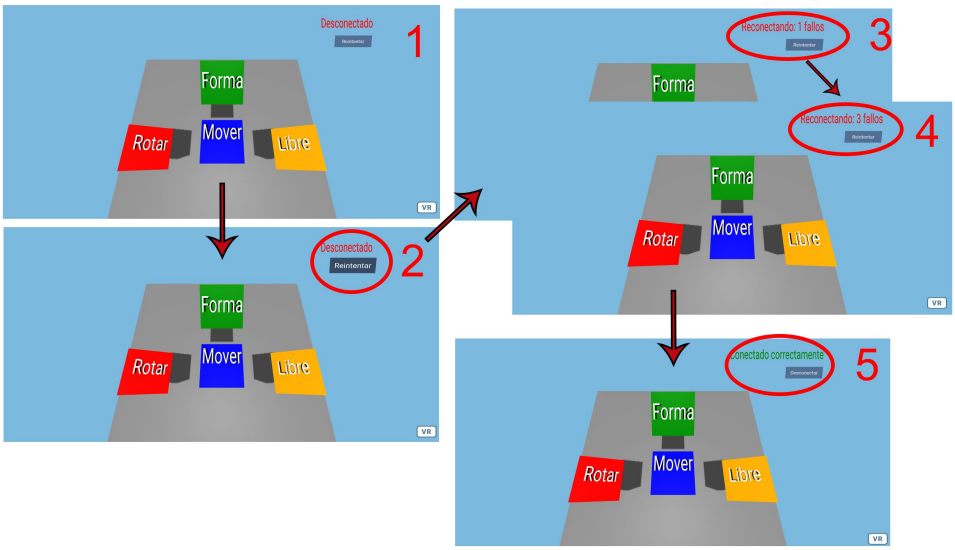


Figura 6.4: Conectar al objeto tangible con éxito

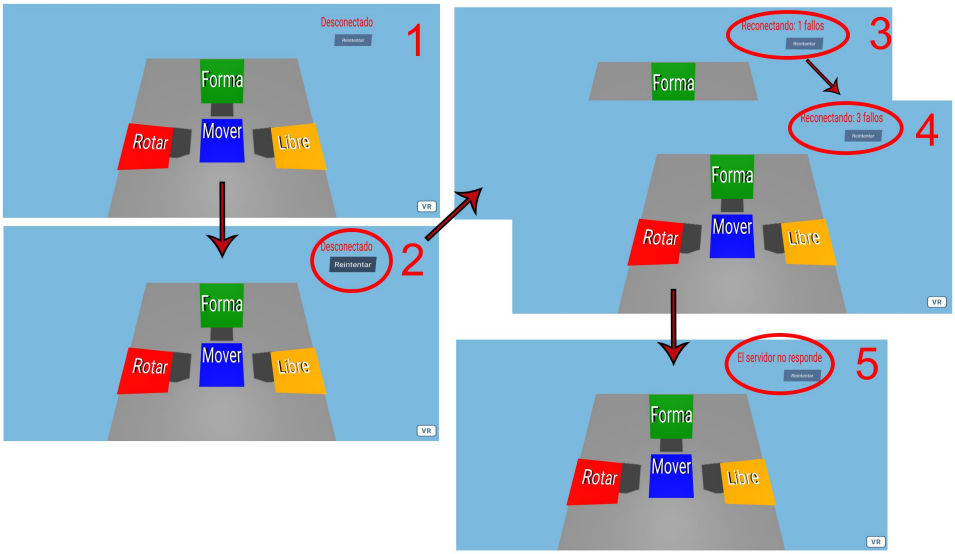


Figura 6.5: Conectar al objeto tangible y fallar

3) Intentar entrar en Rotar, Mover, Libre

Basado en el caso de uso 3.5. El usuario puede poner el ratón sobre los cubos de Rotar, Mover y Libre y verá que reaccionan haciendo una animación de agrandarse, pero como se espera que ocurra, si pulsa en alguno de ellos y está desconectado, no accederá a ninguna de esas escenas.

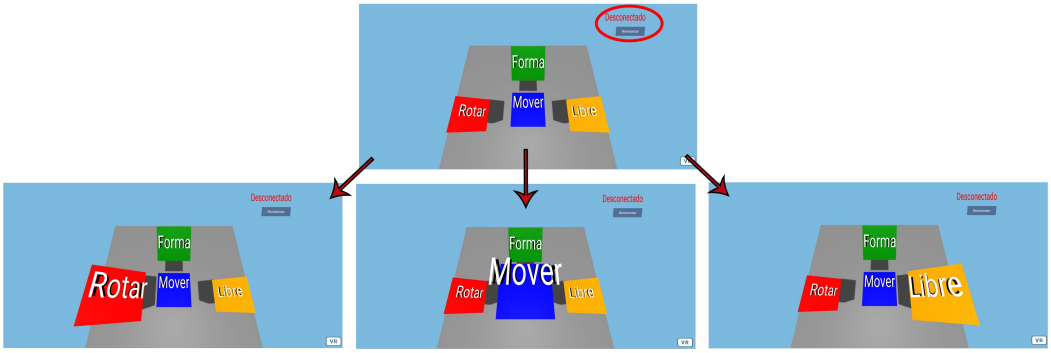


Figura 6.6: Intentar entrar en Rotar, Mover, Libre estando desconectado

4) Entrar en Forma

Basado en el caso de uso 3.6. En esta ocasión, el usuario puede acceder tanto si se está desconectado del objeto tangible como se ve en la parte superior derecha de la pantalla *1* en la izquierda de la figura 6.7 o conectado como se ve en la pantalla *1* de la derecha de la figura mencionada. En ambas rutas, se espera que al poner el ratón sobre el cubo Forma, éste se agrande y, como en la pantalla *2*, al pulsar sobre el mismo, se acerque y se agrande hasta llegar a ocupar casi la pantalla en su totalidad y cambiar a la escena de Forma en la pantalla *3*.

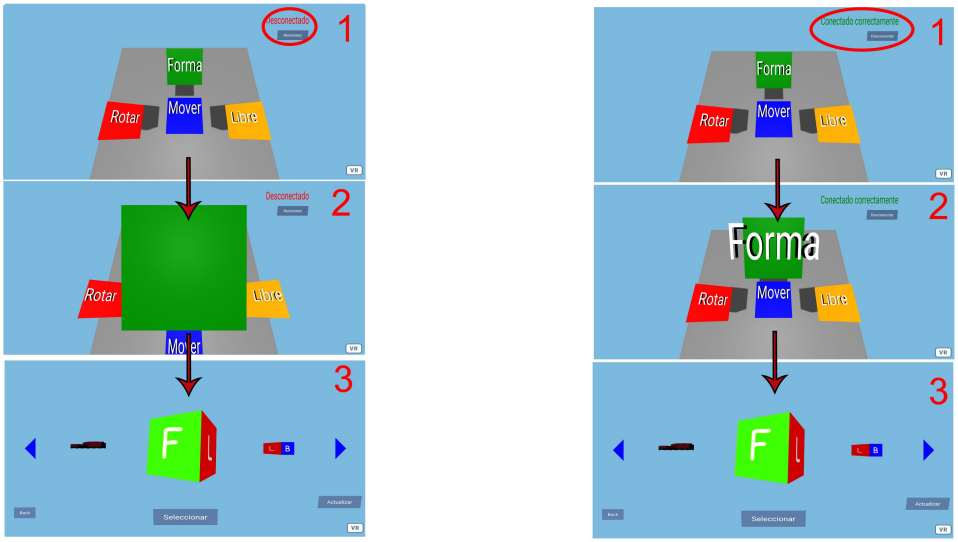


Figura 6.7: Entrar en Forma: desconectado a la izquierda y conectado a la derecha

5) Entrar en Rotar

Basado en el caso de uso 3.12. Estando conectado se espera poder entrar en la escena de Rotar. En la figura 6.8, desde la pantalla *1*, al poner el ratón sobre el cubo Rotar, este se agranda como en la pantalla *2* y, al pulsarlo, hará una animación acercándose hasta ocupar todo el espacio en la pantalla *3* y cambiar a la escena de Rotar en la pantalla *4*.

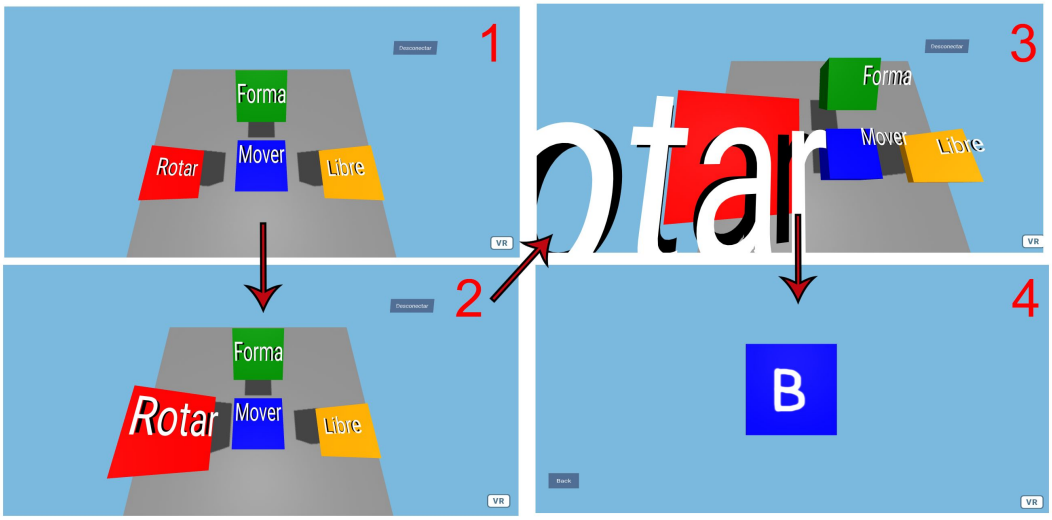


Figura 6.8: Entrar en Rotar estando conectado

6) Entrar en Mover

Basado en el caso de uso 3.17. Estando conectado se espera poder entrar en la escena de Mover. En la figura 6.9, desde la pantalla *1*, al poner el ratón sobre el cubo Mover, este se agranda como en la pantalla *2* y, al pulsarlo, hará una animación acercándose hasta ocupar todo el espacio en la pantalla *3* y cambiar a la escena de Rotar en la pantalla *4*.

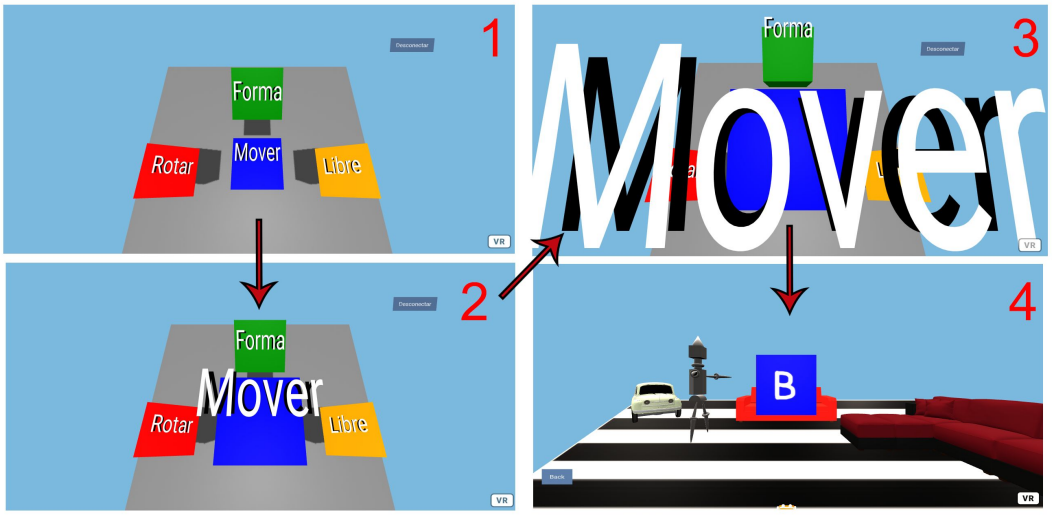


Figura 6.9: Entrar en Mover estando conectado

7) Entrar en Libre

Basado en el caso de uso 3.25. Estando conectado se espera poder entrar en la escena de Libre. En la figura 6.10, desde la pantalla *1*, al poner el ratón sobre el cubo Libre, este se agranda como en la pantalla *2* y, al pulsarlo, hará una animación acercándose hasta ocupar todo el espacio en la pantalla *3* y cambiar a la escena de Rotar en la pantalla *4*.

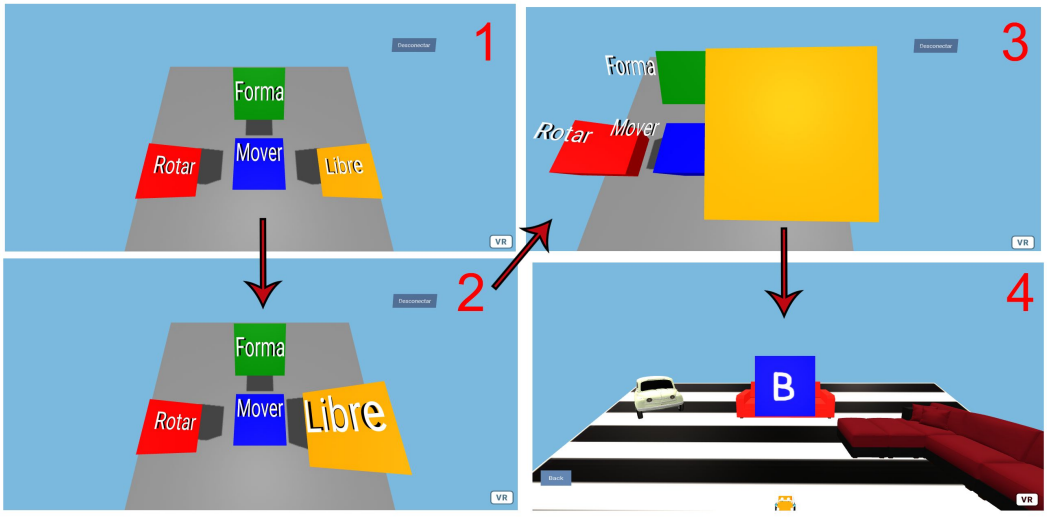


Figura 6.10: Entrar en Libre estando conectado

6.3. Forma

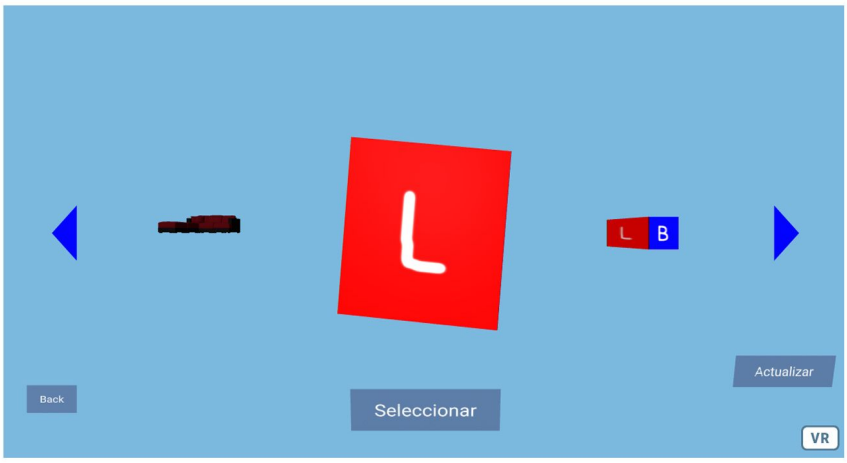


Figura 6.11: Escena de Forma

En esta escena se encuentra la gestión y selección de las formas que puede tomar el objeto tangible de HITO. En esta escena se encuentran los siguientes elementos:

- **Carrusel:** elemento que se encuentra en el centro de la escena y consta de un objeto central más grande y otros dos, uno a cada lado del centra, más pequeños.

  Objeto central: es la forma que tomará el objeto tangible si se pulsa el botón Seleccionar. Este objeto rotará por sí mismo con una pequeña angulación y es pulsable. Si se mantiene la pulsación sobre el objeto y se mueve el ratón por la pantalla, el objeto rotará con el movimiento del ratón.

  Objetos secundarios: no son interactuables, pero representan la forma que tomará el objeto central cuando se pulsen las flechas de dirección izquierda o derecha.

- **Flecha izquierda:** una flecha que se pone roja cuando se pone el cursor sobre ella y que cuando es pulsada, hace que el carrusel gire hacia la izquierda y mueve el objeto central se ponga a su izquierda (derecha para el usuario) y se coloque el objeto secundario de la izquierda en el centro.

- **Flecha derecha:** una flecha que se pone roja cuando se pone el cursor sobre ella y que cuando es pulsada, hace que el carrusel gire hacia la derecha y mueve el objeto central se ponga a su derecha (izquierda para el usuario) y se coloque el objeto secundario de la derecha en el centro.

- **Seleccionar:** resalta al ponerse el ratón sobre él. Cuando se pulsa, cambia a la escena 6.2 y tanto el objeto tangible como los cubos del menú toman la forma seleccionada.

- **Actualizar:** botón situado abajo a la derecha de la escena que resalta cuando se pone el cursor encima. Su función es la de actualizar el objeto tangible dependiendo de los cambios en el directorio *'assets/models/'*. Si no se ha hecho nada, no se actualizará el carrusel, pero si se ha añadido o quitado un archivo de modelo en ese directorio, se añadirá o quitará respectivamente el objeto en cuestión. En el manual de usuario en el anexo D se explica más a fondo la gestión de los objeto

- **Back:** al poner el ratón sobre él resaltará y al pulsarlo llevará a la escena 6.2 del menú principal.

Esta escena consta de las siguientes experimentaciones basadas en los casos de uso:

1) Rotar carrusel

Basado en el caso de uso 3.7. En la escena de Forma, el cambio producido al pulsar las flechas de izquierda y derecha, tiene que ser el de cambiar la forma del objeto grande central por el de su izquierda o el de su derecha.

Para rotar el carrusel hacia la izquierda y llevar el objeto de la izquierda al centro, hay que seguir la ruta izquierda de la figura 6.12. Desde la pantalla *1*, cuando el usuario pone el ratón sobre la flecha izquierda, esta reacciona y se pone roja como en la pantalla *2*. Observando el objeto rodeado del circulo amarillo de esa misma pantalla, al pulsar la flecha, esta hará una pequeña animación y se pasará a la pantalla *3*, lo que

hará que ese objeto pase a ser el central. Si se vuelve a pulsar, se pasará a la pantalla *4*, donde vuelve a cambiar objeto central.

Para rotar el carrusel hacia la derecha y llevar el objeto de la derecha al centro, hay que hacer lo mismo que para la izquierda, pero pulsando la flecha de la derecha.

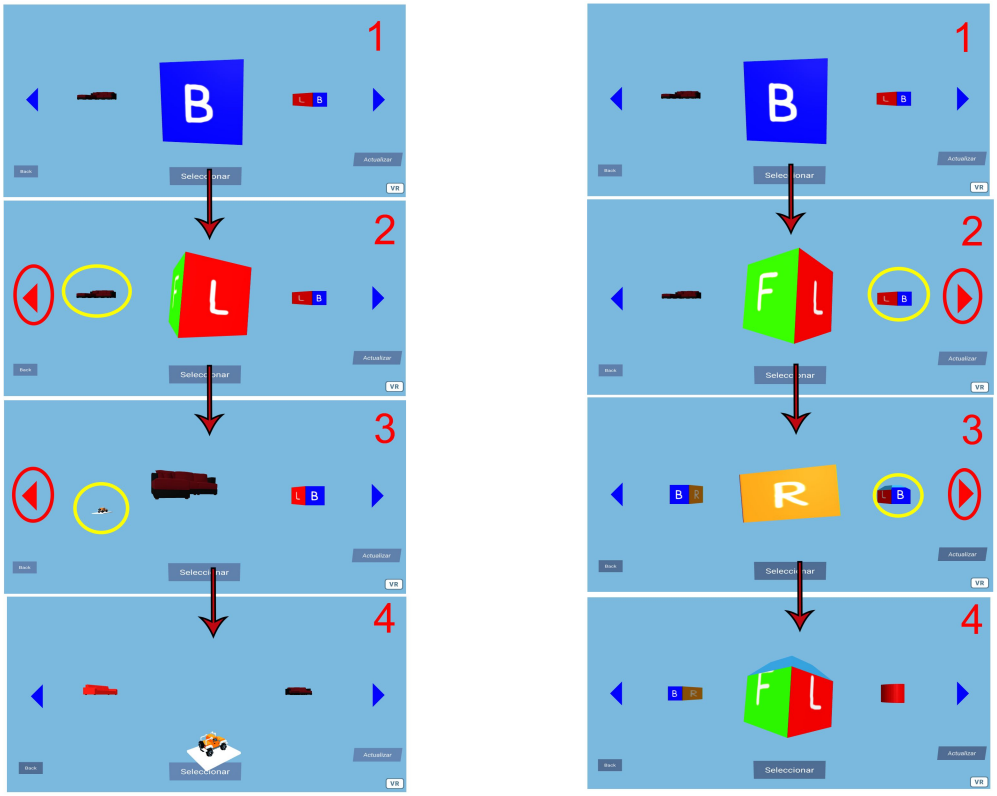


Figura 6.12: Rotar carrusel: hacia la izquierda en la izquierda y hacia la derecha en la derecha

2) Actualizar carrusel

Basado en el caso de uso 3.8. Se espera que ocurran tres resultados.

El primero de ellos es el de la figura 6.13. Empezando en la primera pantalla, cuando el ratón se encuentra sobre el botón *Actualizar*, este reacciona haciéndose más grande como en la segunda pantalla. Al pulsarlo, hará una pequeña animación y se pasará a la tercera, en la que nada habrá ocurrido porque no se ha modificado el

directorio donde se encuentran los archivos de las formas de los objetos que se ven.

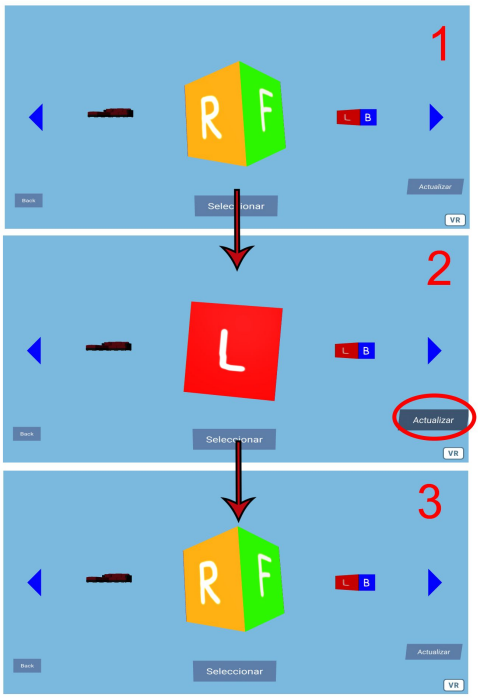


Figura 6.13: Actualizar carrusel y no ocurre nada

El segundo ocurre cuando alguno de los archivos de los modelos es movido de la carpeta *'/assets/models/'*. Estando la interfaz de en la pantalla *1* de la izquierda de la figura 6.14, se eliminar los archivos de un modelo como se ven en las pantallas *2* y *3*. En la *4*, poniendo el ratón sobre el botón actualizar, éste hará una animación y si lo pulsamos, se pasará a la pantalla *5*, observando que el objeto en un círculo amarillo en la pantalla *4*, ha desaparecido.

El tercer caso es el inverso del segundo y ocurre cuando se añaden archivos de modelos de formas en la carpeta. En la parte derecha de la figura 6.14, cuando se está en la pantalla *1*, al añadir los objetos de las pantallas *2* y *3* y pulsar actualizar en la pantalla *4*, el objeto aparecerá.

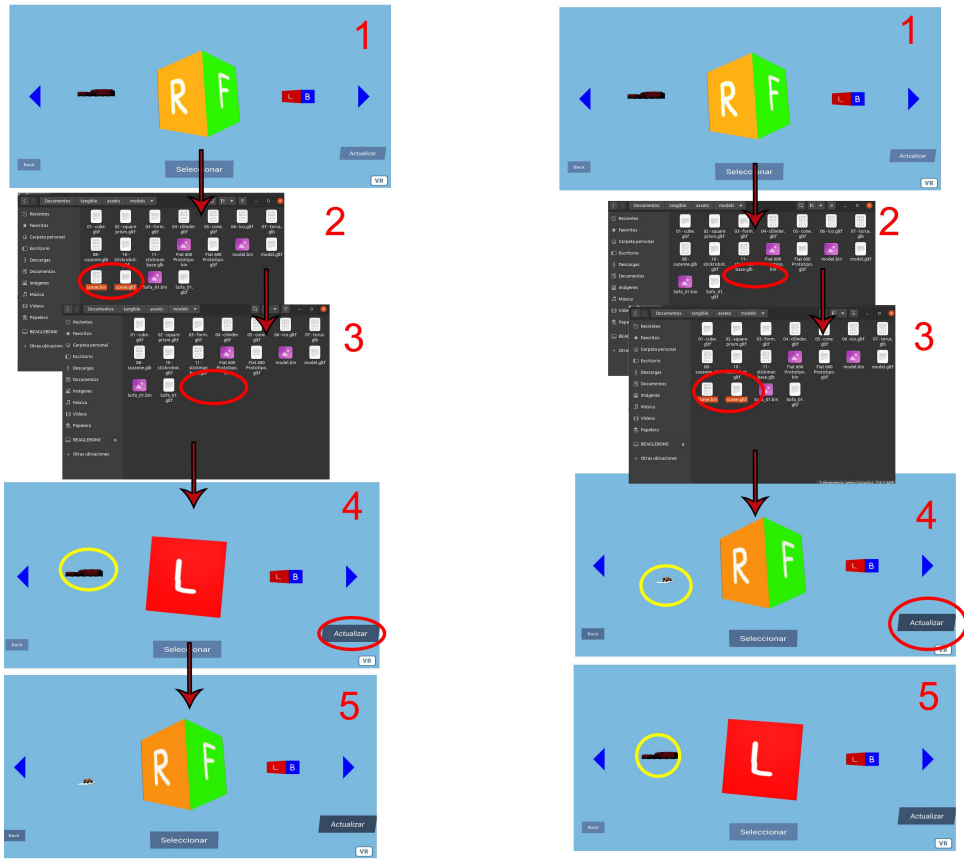


Figura 6.14: Actualizar carrusel: se borra un objeto a la izquierda y se añade a la derecha

3) Rotar objeto

Basado en el caso de uso 3.9. Al mantener el ratón pulsado sobre el objeto central, éste rota como se esperaba. Si se suelta el objeto, vuelve a rotar por su cuenta.

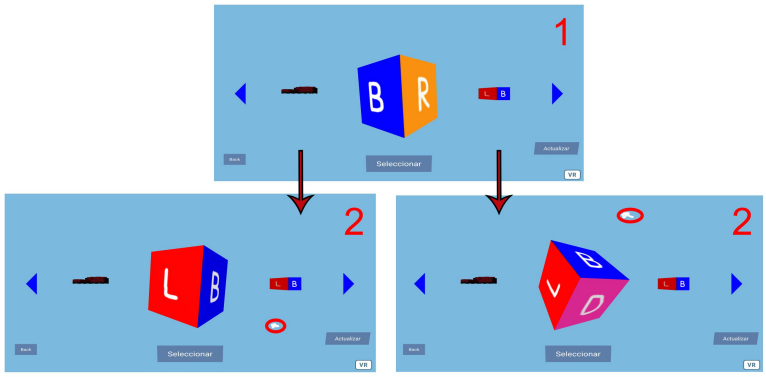


Figura 6.15: Rotar objeto central con el ratón

4) Salir de forma

Basado en el caso de uso 3.10. Estando en la pantalla *1* de la figura 6.16, se pasa a la pantalla *2* al poner el ratón sobre el botón *'Back'*, que reacciona y hace una animación de agrandarse. Al pulsar en él, se pasa a la pantalla *3* donde el cubo de Forma se aleja desde la posición en la que se había acercado en la experimentación **3**) hasta volver colocarse todo en la posición original del menú principal en la pantalla *4*.

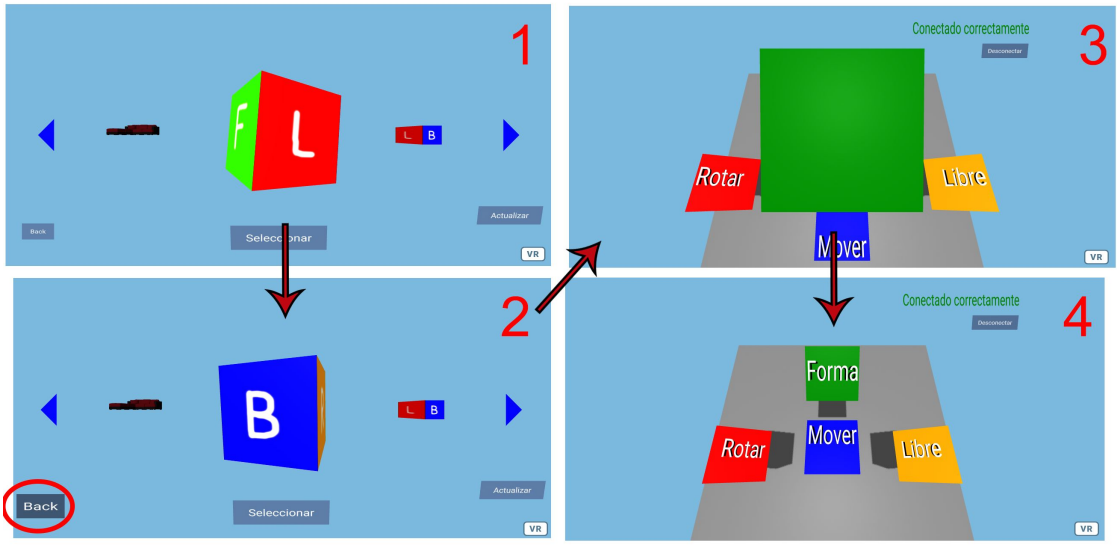


Figura 6.16: Salir de Forma

5) Cambiar forma al objeto tangible

Basado en el caso de uso 3.11. Se espera que al seleccionar la forma, tanto el menú como el objeto tangible cambie. En la figura 6.17, en la pantalla *1* se tiene una forma concreta y al poner el ratón sobre el botón seleccionar, éste se agranda como se ve en la pantalla *2*. Al pulsarlo hará una animación y se pasará a la pantalla *3*, haciendo una animación hasta llegar a la pantalla *4*, que es como estaba colocado al principio el menú principal.

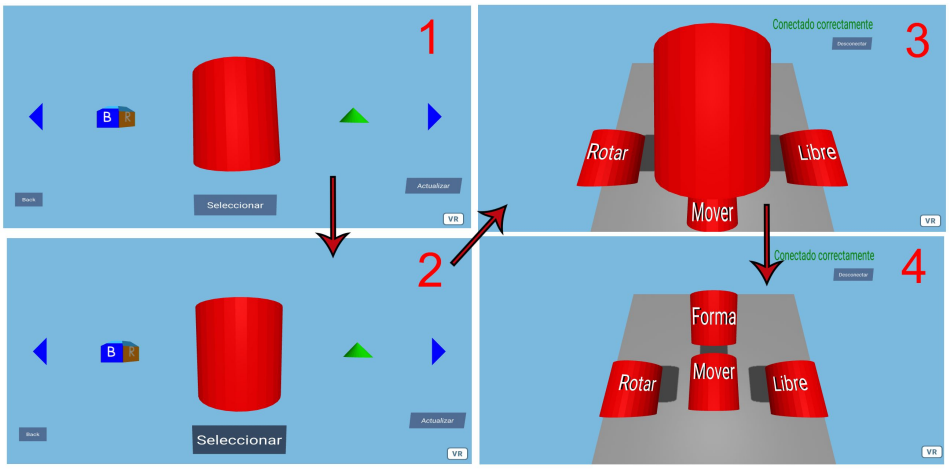


Figura 6.17: Seleccionar forma y cambiar el objeto y el menú principal

Para verificar el resultado de haber cambiado el objeto tangible, en la figura 6.18 se pulsa en Rotar en la pantalla *1* y en la *2* se ve que el objeto ha cambiado.

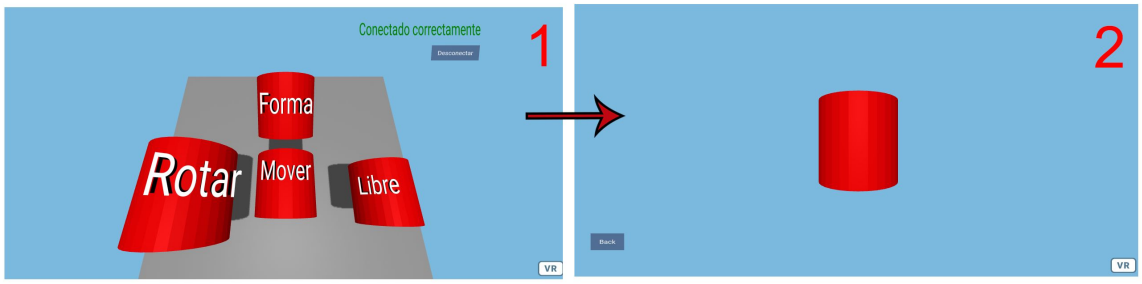


Figura 6.18: Comprobar que al cambiar la forma se cambia el objeto tangible

6.4. Rotar

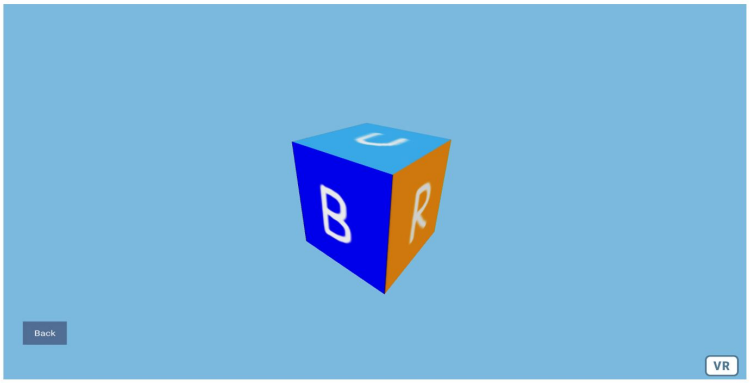


Figura 6.19: Escena de Rotar

En esta escena se muestra el objeto que representa el objeto tangible con la forma inicial o con la seleccionada en la escena 6.3 y un botón en la esquina inferior izquierda en el que pone 'Back'.

- **Objeto tangible:** la representación del objeto tangible de HITO, hará todas las rotaciones que el usuario haga sobre el objeto físico. Al rotar con el objeto real, se fomenta el entendimiento de la pieza en 3D en la pantalla.
- **Back:** al poner el ratón sobre él resaltará y al pulsarlo llevará a la escena 6.2 del menú principal.

Esta escena consta de las siguientes experimentaciones basadas en los casos de uso:

1) Rotar horizontalmente

Basado en el caso de uso 3.13. En esta experimentación se hace uso de la Beagle-Bone, por lo que es necesario calcular el retardo entre los datos que da y los que se muestran en Ubuntu. En la tabla E.1 se muestra un retraso de 24,06 milisegundos.

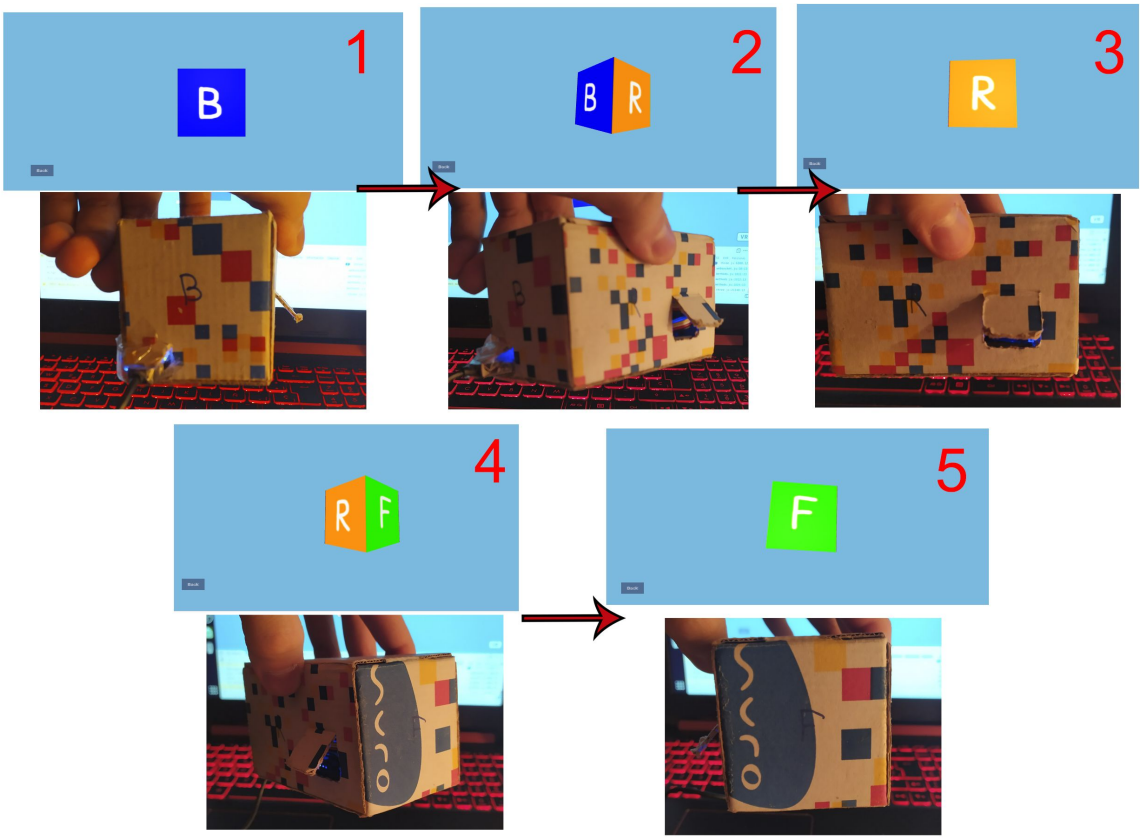
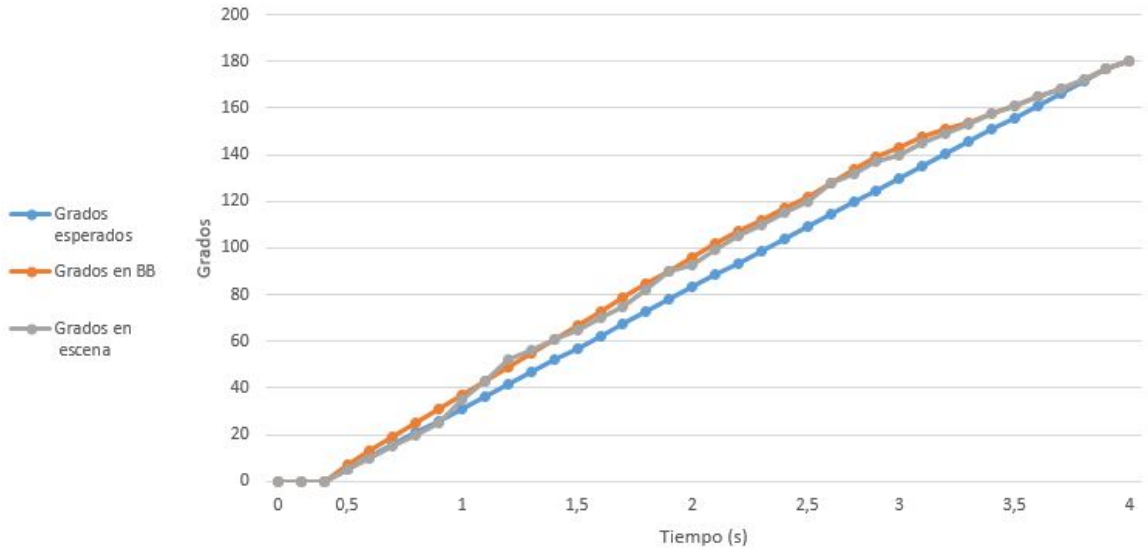


Figura 6.20: Rotación horizontal del objeto tangible e imitación en la interfaz en Rotar

La rotación mostrada en la figura 6.20 donde se ve que al rotar hacia la derecha horizontalmente el objeto tangible, su representación en la interfaz sigue su movimiento a lo largo de las 5 pantallas mostradas. Para comprobar que la rotación es de 180 grados sobre el eje horizontal y que gira a una velocidad uniforme durante 4 segundos como se espera, se muestra la gráfica 6.21.

En la gráfica se muestran las series de datos de los grados que tanto la BeagleBone como el objeto en escena muestran utilizando los valores del eje Z del giroscopio y los valores obtenidos del eje Z del modelo del objeto tangible respectivamente. También se encuentra la serie de los grados esperados para comparar con las otras dos. El movimiento que representa la gráfica es el de la figura 6.20.

6.21: Gráfica de rotación horizontal uniforme en el tiempo en Rotar



Las series de la BB y de la escena se solapan en muchos puntos y resultan estar un poco por encima de los valores esperados. Se observa que al mover el objeto tangible a una velocidad uniforme, el aumento de los grados es uniforme también. El objeto tangible comienza a rotar a los 0.5 segundos, por lo que los datos anteriores corresponden a pantalla *1* de la figura 6.20. Cerca de 1 segundo después, se alcanzan los 45º como se muestra en la pantalla *2* y, llegados a los 2 segundos, se alcanzan los 90º y la pantalla *3*. Siguiendo el movimiento, a los 3 segundos más o menos, se alcanzan los 135º que se muestran en la pantalla *4* y termina con el giro de 180º con la pantalla *5*.

Aún estando un poco por encima de los valores esperados, las series tienen una inclinación similar y varían al mismo ritmo, por lo que se ajusta a lo esperado.

2) Rotar verticalmente

Basado en el caso de uso 3.14. En esta experimentación se hace uso de la BeagleBone, por lo que es necesario calcular el retardo entre los datos que da y los que se muestran en Ubuntu. En la tabla E.2 se muestra un retraso de 26,27 milisegundos.

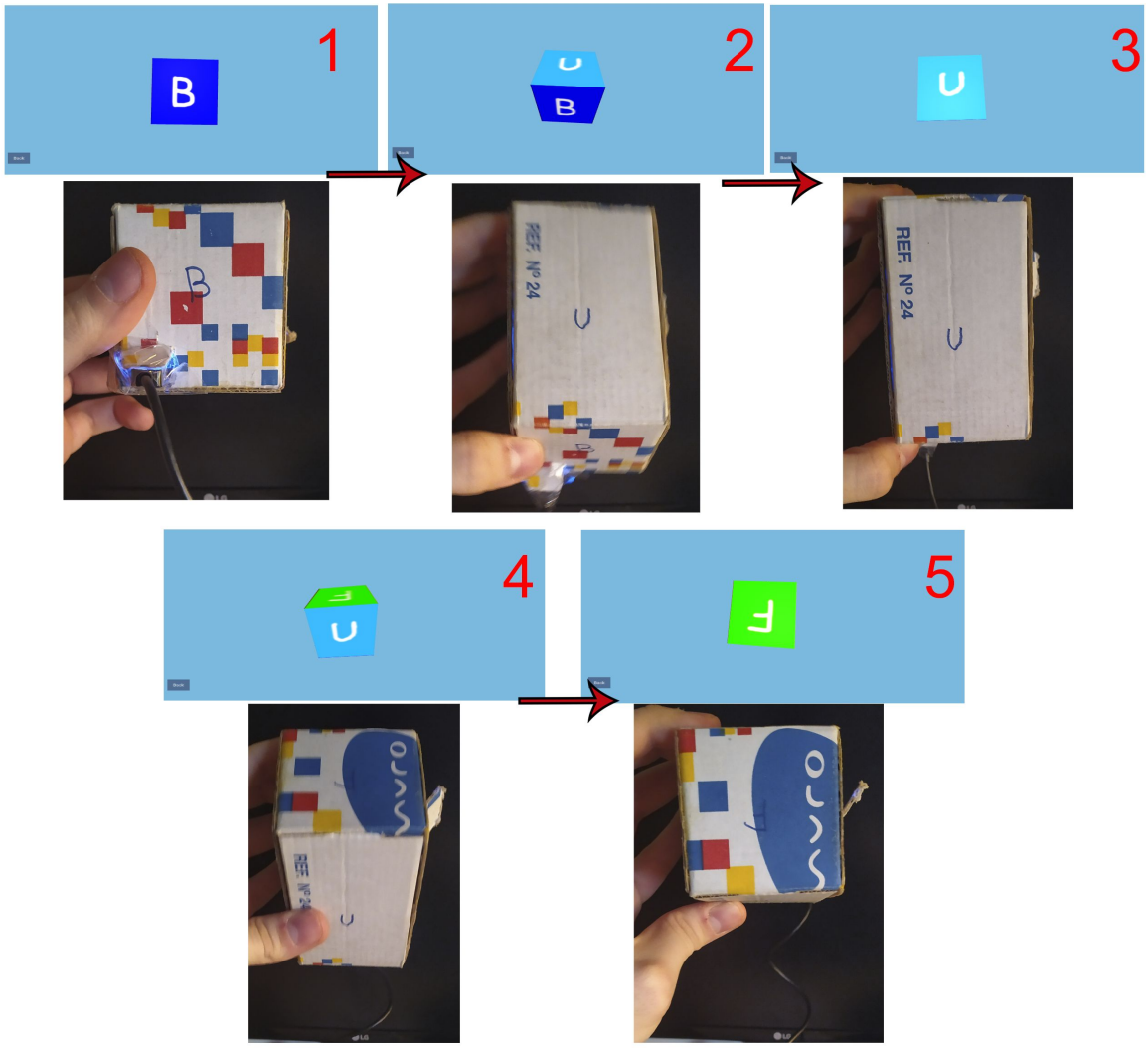


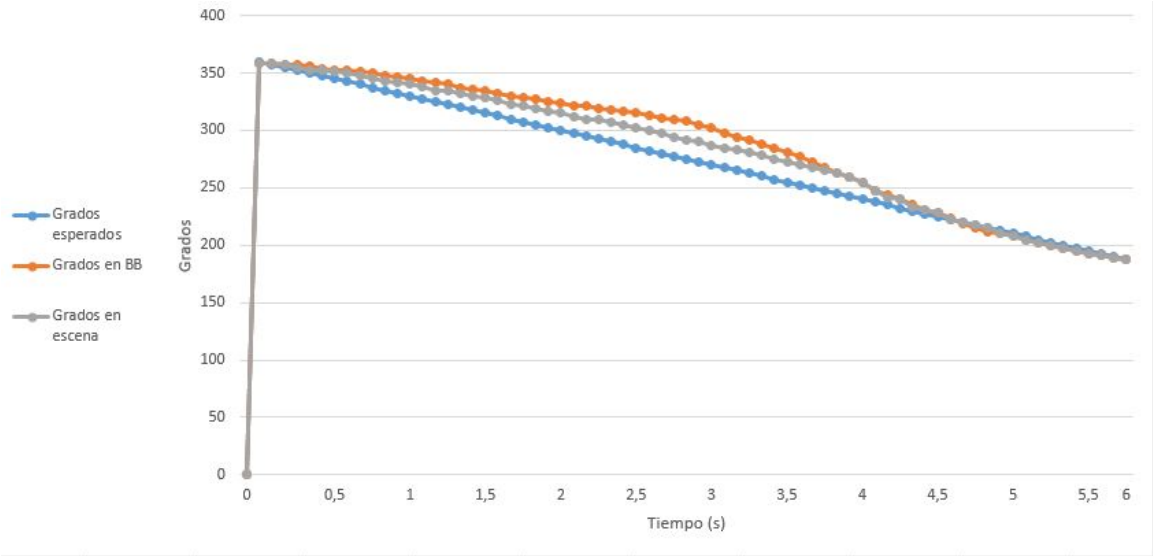
Figura 6.22: Rotación vertical del objeto tangible e imitación en la interfaz en Rotar

La rotación mostrada en la figura 6.22 donde se ve que al rotar hacia la arriba verticalmente el objeto tangible, su representación en la interfaz sigue su movimiento a lo largo de las 5 pantallas mostradas. Para comprobar que la rotación es de 180 grados sobre el eje vertical y que gira a una velocidad lo más uniforme posible durante 6 segundos como se espera, se muestra la gráfica 6.23.

En la gráfica se muestran las series de datos de los grados que tanto la BeagleBone como el objeto en escena muestran utilizando los valores del eje X del acelerómetro

transformado a grados y los valores obtenidos del eje X del modelo del objeto tangible respectivamente. También se encuentra la serie de los grados esperados para comparar con las otras dos. El movimiento que representa la gráfica es el de la figura 6.22.

6.23: Gráfica de rotación vertical uniforme en el tiempo en Rotar



Las series de la BB y de la escena se solapan en algunos puntos y resultan estar un poco por encima de los valores esperados. Al ser 0º en el eje horizontal y 270º cuando se inclina 90º hacia atrás, el movimiento es en sentido antihorario y disminuyendo los grados. Se observa que al mover el objeto tangible la velocidad es uniforme en dos partes, desde la pantalla *1* de la figura 6.22 a la *3* cuando el objeto tangible está verticalmente y desde esa posición hasta la mostrada en la pantalla *5* cuando pasa a estar en horizontal. La primera parte corresponde hasta más o menos lo 3,5 segundos, siendo este punto la pantalla *3* y el de los 2 segundos la pantalla *2*. La segunda parte corresponde desde los 3.5 segundos a los 6, estando en los 4.5 segundos más o menos en lo mostrado en la pantalla *4*.

Aún estando un poco por encima de los valores esperados, las series varían al mismo ritmo, siendo el punto intermedio donde se produce la mayor diferencia debido

al cambio de manos al realizar el giro con la BB. Se ajusta a lo esperado contando con esa diferencia.

3) Rotar sobre sí mismo

Basado en el caso de uso 3.15. En esta experimentación se hace uso de la BeagleBone, por lo que es necesario calcular el retardo entre los datos que da y los que se muestran en Ubuntu. En la tabla E.3 se muestra un retraso de 25 milisegundos.

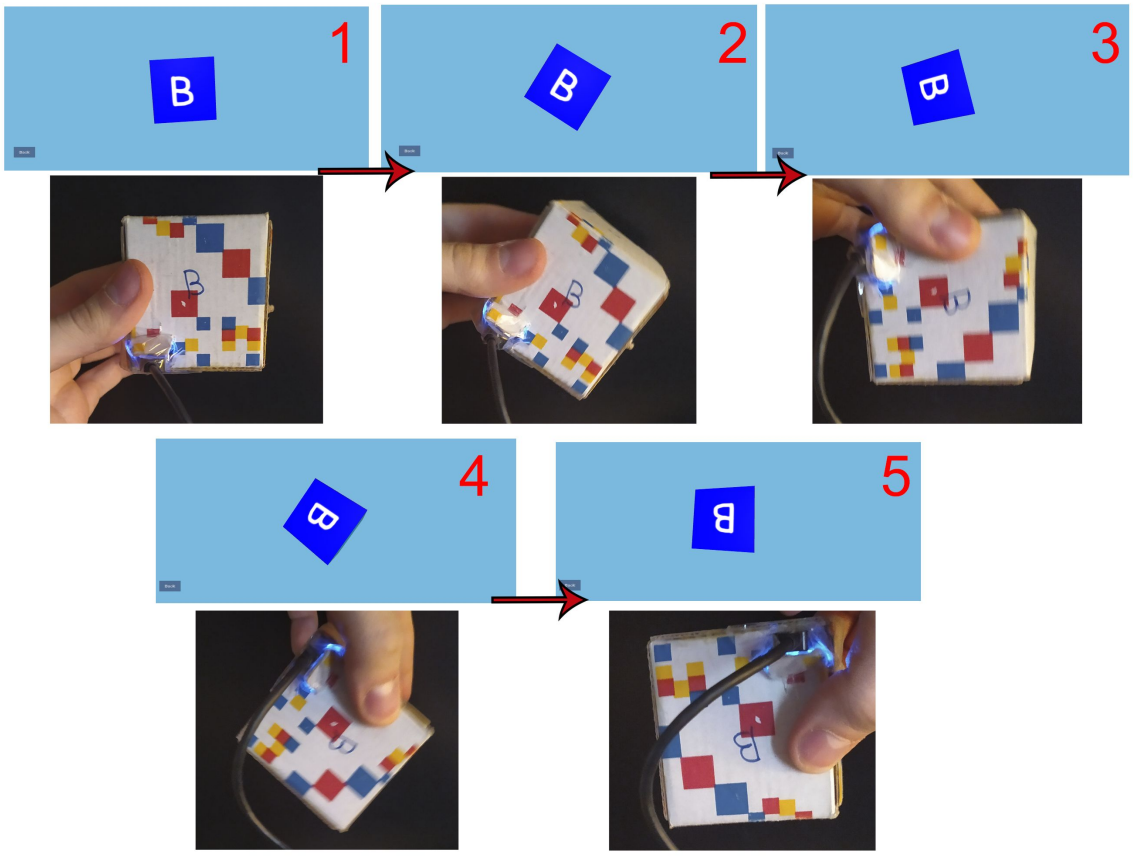


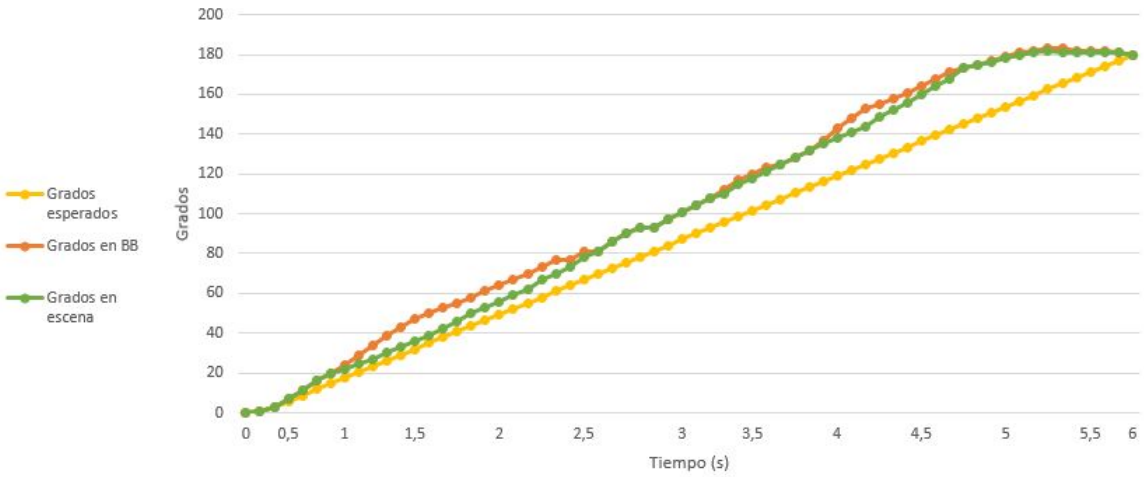
Figura 6.24: Rotación sobre sí mismo del objeto tangible e imitación en la interfaz en Rotar

La rotación mostrada en la figura 6.24 donde se ve que al rotar hacia la derecha sobre sí mismo el objeto tangible, su representación en la interfaz sigue su movimiento a lo largo de las 5 pantallas mostradas. Para comprobar que la rotación es de 180 grados sobre sí mismo y que gira a una velocidad uniforme durante 5,5 segundos como

se espera, se muestra la gráfica 6.25.

En la gráfica se muestran las series de datos de los grados que tanto la BeagleBone como el objeto en escena muestran utilizando los valores del eje Y del acelerómetro transformado a grados y los valores obtenidos del eje Y del modelo del objeto tangible respectivamente. También se encuentra la serie de los grados esperados para comparar con las otras dos. El movimiento que representa la gráfica es el de la figura 6.24.

6.25: Gráfica de rotación uniforme sobre sí mismo en el tiempo en Rotar



Las series de la BB y de la escena se solapan en muchos puntos y resultan estar por encima de los valores esperados. Se observa que al mover el objeto tangible a una velocidad uniforme, el aumento de los grados es uniforme también. El objeto tangible comienza a rotar a los 0.5 segundos, por lo que los datos anteriores corresponden a pantalla *1* la figura 6.24. Al segundo 1,7 más o menos, se alcanzan los 45º como se muestra en la pantalla *2* y, llegados a los 3 segundos, se alcanzan los 90º y la pantalla *3*. Siguiendo el movimiento, a los 4.3 segundos más o menos, se alcanzan los 135º que se muestran en la pantalla *4* y termina con el giro de 180º con la pantalla *5* a los 5.5 segundos.

Aunque los valores están por encima de los esperados, tienen una inclinación similar y terminan con los mismos grados. El resultado. Se ajusta al rango de error de lo esperado.

4) Salir de rotar

Basado en el caso de uso 3.16. Desde la escena de Rotar como se muestra en la pantalla *1* de la figura 6.26, el usuario puede volver al menú principal. Para ello, al poner el ratón sobre el botón *'Back'* como en la pantalla *2*, éste botón se hará grande. Al pulsarlo, hará una animación y pasará a la escena del menú principal en la pantalla *3*, haciendo la animación inversa que cuando se entró en la escena Rotar, ocupando todo como en la pantalla *3* hasta llegar al menú principal en la pantalla *5*, pasando por la pantalla *4*.

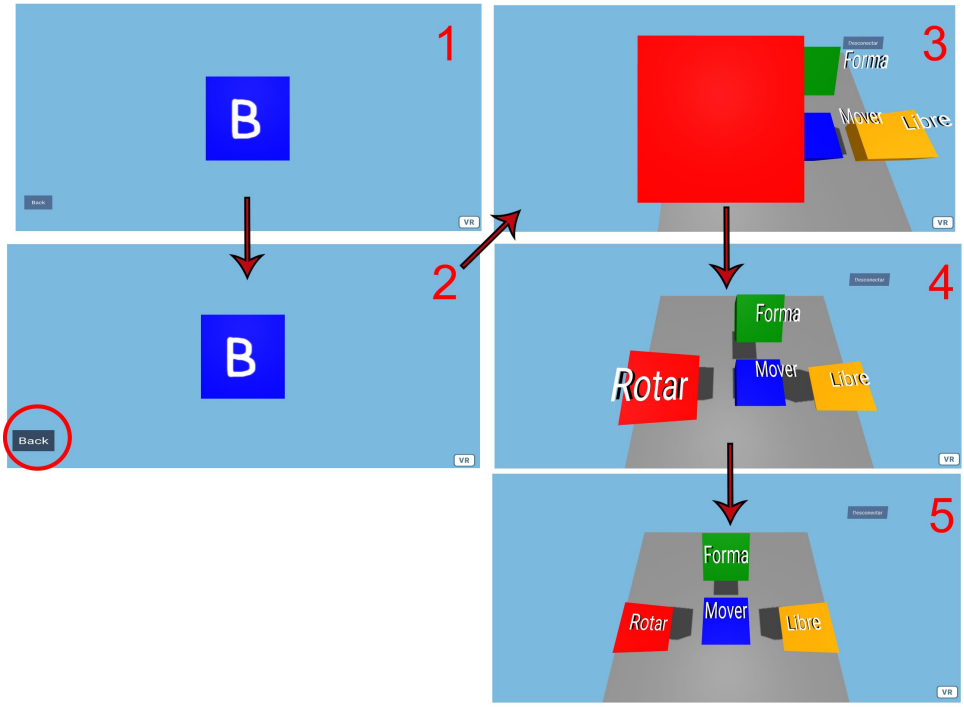


Figura 6.26: Salir de Rotar

6.5. Mover

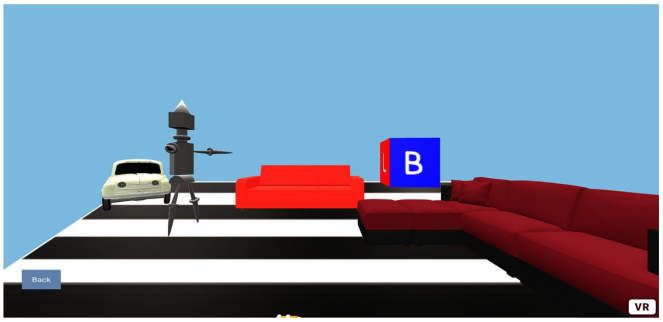


Figura 6.27: Escena de Mover

En esta escena se muestra el objeto que representa el objeto tangible con la forma inicial o con la seleccionada en la escena 6.3, un botón en la esquina inferior izquierda en el que pone 'Back' y una escena con diversos objetos interactuables.

- **Objeto tangible:** la representación del objeto tangible de HITO. En esta escena rotará levemente pero estará enfocado en desplazarse por toda la escena, moviéndose hacia la izquierda cuando el objeto físico rota hacia la izquierda y viceversa. También pasa con las rotaciones posteriores y frontales que acercará y alejará el objeto del punto de vista. Si se inclina mucho posterior o frontalmente, el objeto ascenderá o descenderá respectivamente. La cámara se sigue al objeto tangible si éste se acerca mucho a los bordes del campo de visión. Al poder desplazarse por toda la escena y existir otros objetos en el escena a distintas distancias, se puede apreciar que el entorno tridimensional tiene profundidad y que existe un espacio dentro.

- **Objetos interactuables:** los objetos de la escena son iteractuables, cuando se logra hacer que el objeto tangible toque otro, éste hará algo dependiendo de qué objeto sea. Algunos muestran un texto con lo que son como el chaise longue y otros se retiran, como el robot.

- **Back:** al poner el ratón sobre él resaltará y al pulsarlo llevará a la escena 6.2 del menú principal.

Esta escena consta de las siguientes experimentaciones basadas en los casos de uso:

1) Mover lateralmente

Basado en el caso de uso 3.18. En esta experimentación se hace uso de la Beagle-Bone, por lo que es necesario calcular el retardo entre los datos que da y los que se muestran en Ubuntu. En la tabla E.4 se muestra un retraso de 25 milisegundos.

Se busca comprobar que cuando el objeto tangible tiene una rotación pequeña sobre sí mismo hacia la izquierda o hacia la derecha, el desplazamiento ocurre lentamente y que, cuando la rotación es mayor a cierto ángulo, el desplazamiento lateral es más rápido. El objeto físico solo rota, no se desplaza. En la figura 6.28 se muestra como sería un desplazamiento lento y en la figura 6.29 uno rápido además de mostrar el movimiento del objeto tangible físico.

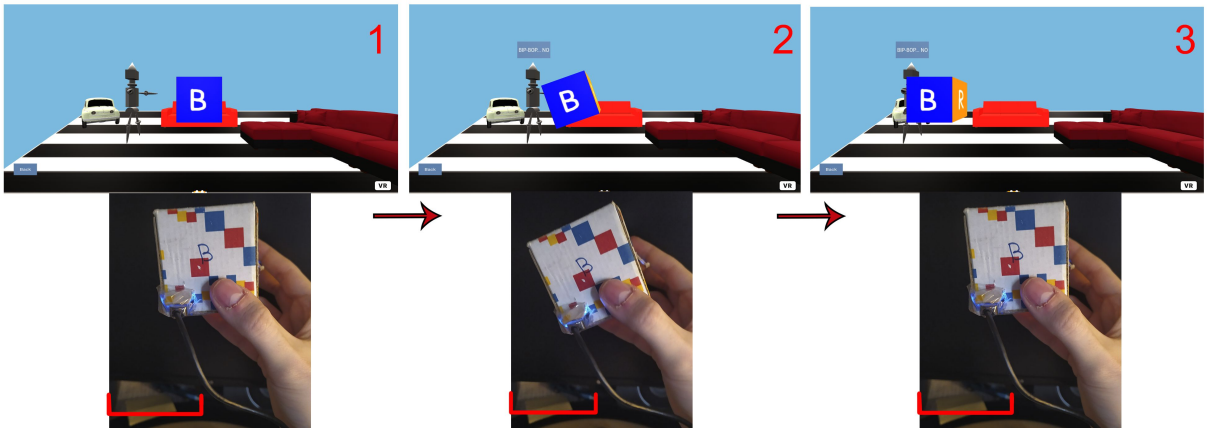


Figura 6.28: Mover lateralmente despacio en Mover

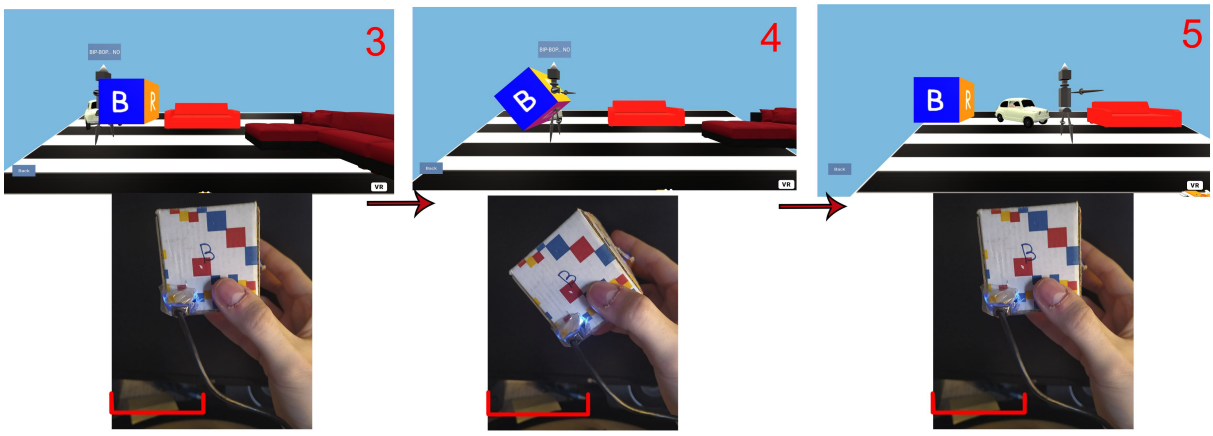
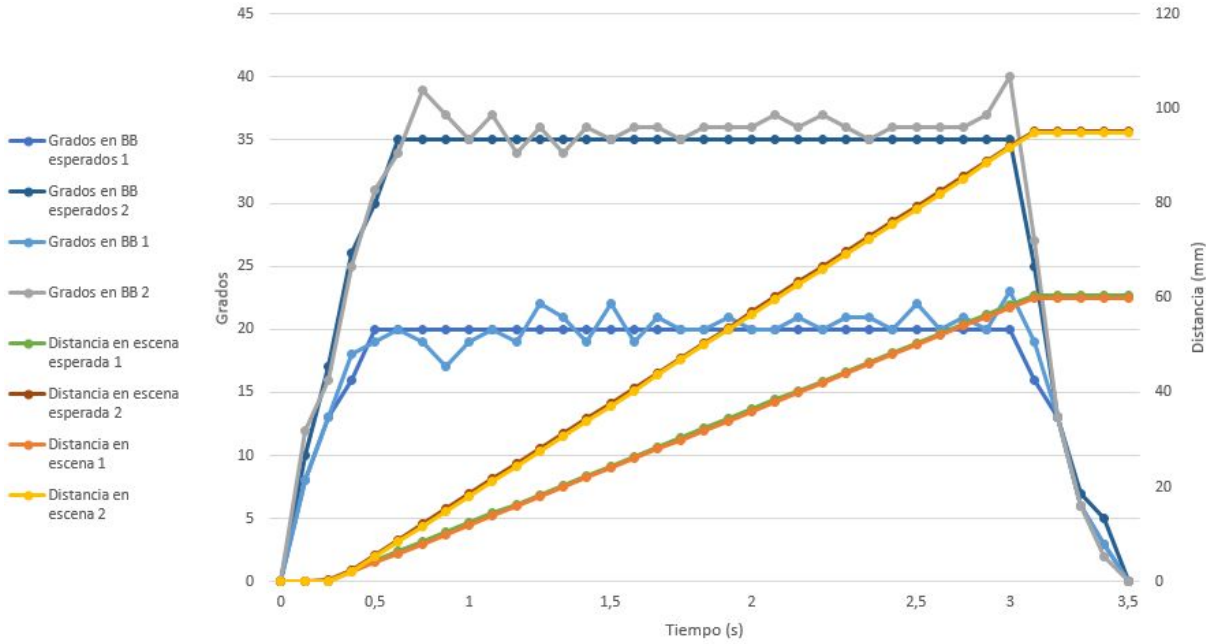


Figura 6.29: Mover lateralmente rápido en Mover

Para comprobar que ocurre lo que se espera, que cuando se rota con un ángulo pequeño, entre 15º y 26º, se mueve más lentamente y cuando es mayor, entre 27º y 55º, la velocidad es mayor, se utiliza la gráfica 6.30 con ambos movimientos representados de las figuras 6.28 y 6.29 respectivamente en un rango de 3,5 segundos.

6.30: Gráfica del desplazamiento lateral en Mover



Las series indican los grados que se obtienen del eje Y de la BB del acelerómetro

al girarla y la distancia recorrida por el modelo en pantalla. De ambos valores se muestran los esperados.

Cuando se realiza una inclinación pequeña, representado este movimiento con el nombre de la serie y un 1, para un desplazamiento más lento del objeto de la escena, los grados obtenidos se mueven en torno a los esperados, mientras que la distancia es muy similar a la esperada. La inclinación más alta se representa con el nombre de la serie y un 2 y se obtiene que los grados obtenidos se encuentran alrededor de los esperados y la distancia recorrida es muy similar a la esperada. La distancia es medida en milímetros en la pantalla física del ordenador, resultando en lo que se ve en la figura 6.31, siendo el 1 de esta figura la posición inicial, el 2 cuando el desplazamiento es lento y el 3 cuando es rápido. En la gráfica se muestra que cuanto más grados, mayor es el desplazamiento, tal y como se esperaba.

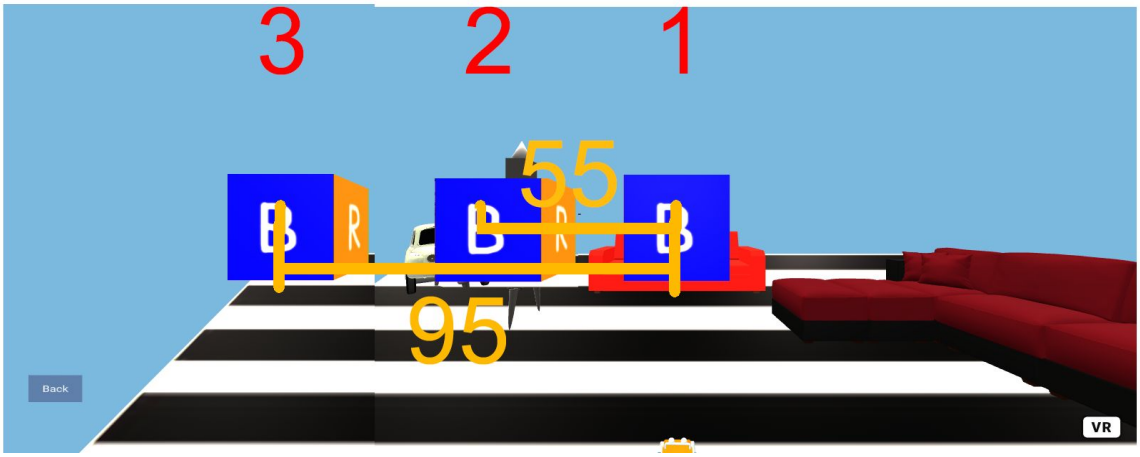


Figura 6.31: Distancia recorrida en el desplazamiento lateral en Mover

La inclinación de las series que se refieren a la distancia tienen la misma inclinación e inician y terminan en el mismo punto cuando los grados disminuyen de los valores mencionados anteriormente. Las series de los grados se ajustan se mueven en torno a los valores esperados.

2) Mover frontalmente

Basado en el caso de uso 3.19. En esta experimentación se hace uso de la Beagle-Bone, por lo que es necesario calcular el retardo entre los datos que da y los que se muestran en Ubuntu. En la tabla E.5 se muestra un retraso de 27 milisegundos.

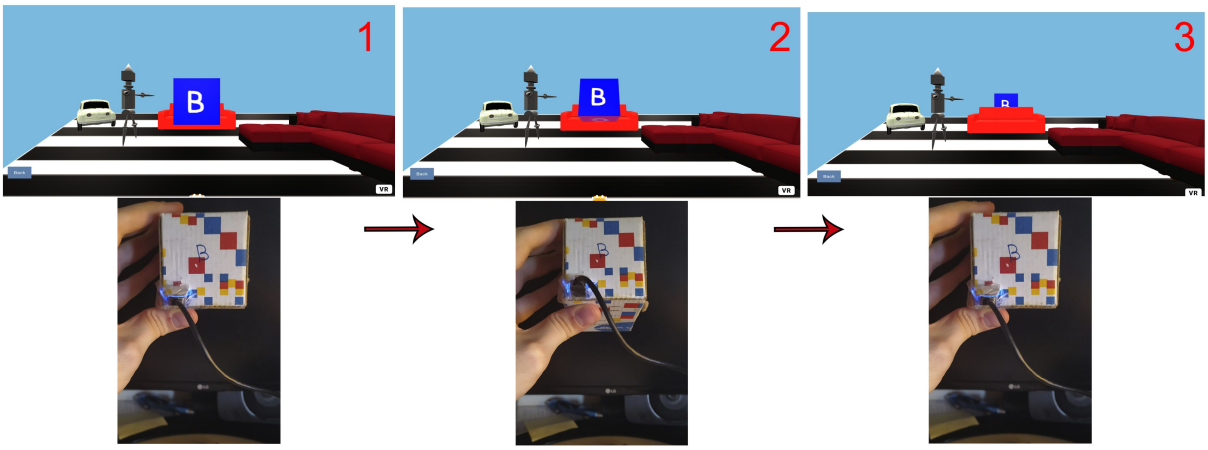


Figura 6.32: Mover frontalmente despacio en Mover

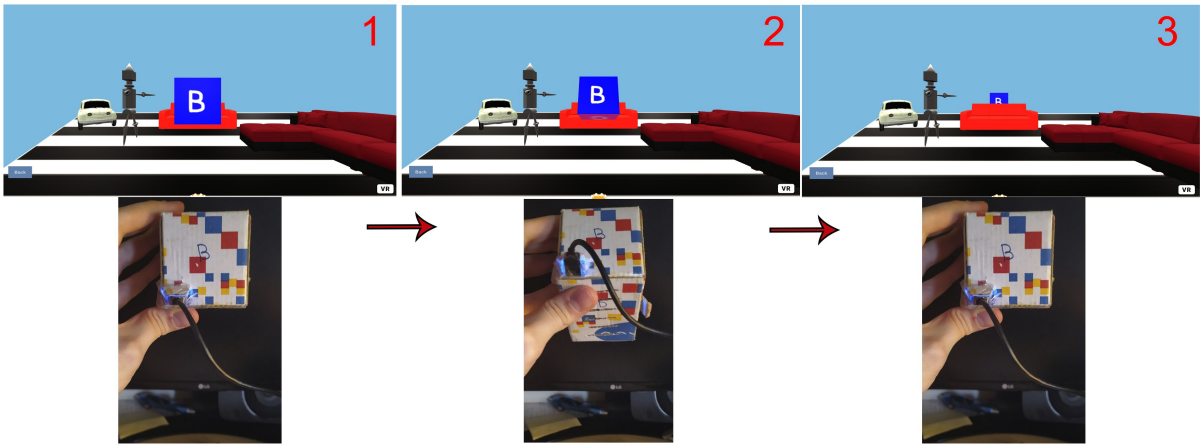


Figura 6.33: Mover frontalmente rápido en Mover

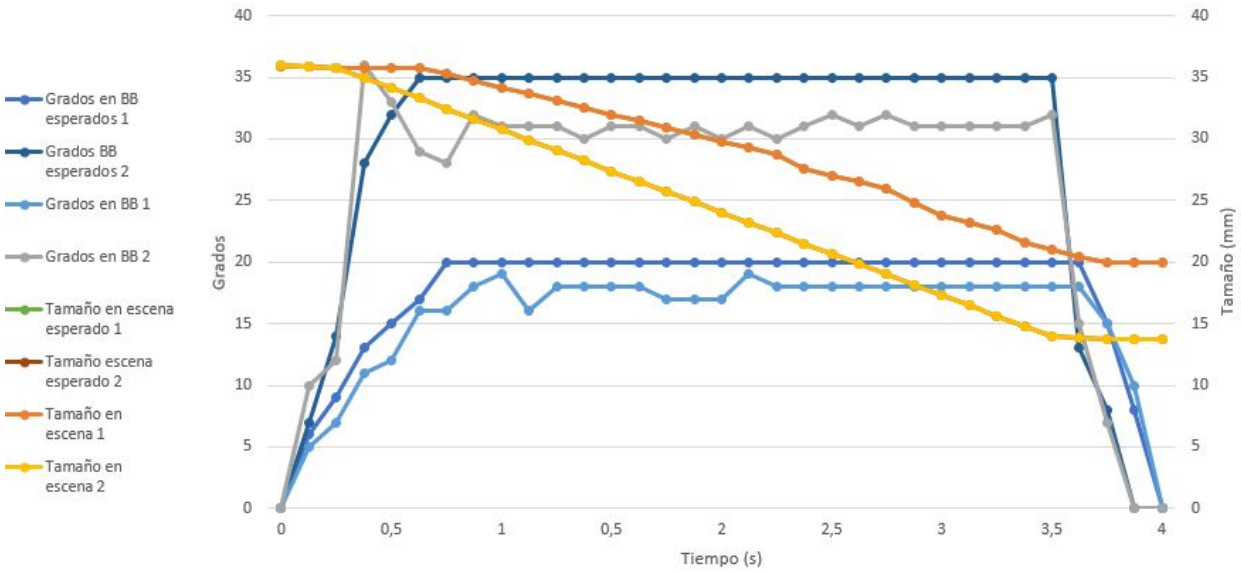
Se busca comprobar que cuando el objeto tangible tiene una rotación pequeña hacia adelante, el desplazamiento ocurre lentamente y que, cuando la rotación es mayor a cierto ángulo, el desplazamiento frontal es más rápido. En la figura 6.32 se muestra

como sería un desplazamiento lento y en la figura 6.33 uno rápido además de mostrar el movimiento del objeto tangible físico.

Para comprobar que ocurre lo que se espera, que cuando se rota con un ángulo pequeño, entre 15º y 26º, se mueve más lentamente y cuando es mayor, entre 27º y 55º, la velocidad es mayor, se utiliza la gráfica 6.37 con ambos movimientos en un rango de 4 segundos.

Las series indican los grados que se obtienen del eje X del acelerómetro de la BB al girarla y el tamaño es el tamaño del modelo del objeto tangible que se ve en la pantalla.

6.34: Gráfica del tamaño del objeto tangible al desplazarse frontalmente en Mover



Para la inclinación pequeña se representa con el nombre de la serie y un 1 y para la inclinación alta con el nombre de la serie y un 2. Con la inclinación pequeña, los grados obtenidos de la BB se encuentran en torno a los esperados, mientras que el tamaño disminuye de forma muy similar al esperado. Con la inclinación más alta, se obtiene que los grados obtenidos se encuentran alrededor de los esperados y el tamaño del

modelo disminuye de forma muy similar al esperado. El tamaño se mide en milímetros en la pantalla física del ordenador, resultando en lo que se ve en las figuras 6.35 y 6.36 junto a la BB.

La inclinación de las series referentes al tamaño es igual a las series esperadas y los grados obtenidos están cerca de las series esperadas.

3) Mover posteriormente

Basado en el caso de uso 3.20. En esta experimentación se hace uso de la BeagleBone, por lo que es necesario calcular el retardo entre los datos que da y los que se muestran en Ubuntu. En la tabla E.6 se muestra un retraso de 10 milisegundos.

Se busca comprobar que cuando el objeto tangible tiene una rotación pequeña hacia atrás, el desplazamiento ocurre lentamente y que, cuando la rotación es mayor a cierto ángulo, el desplazamiento posterior es más rápido. En la figura 6.35 se muestra como sería un desplazamiento lento y en la figura 6.36 uno rápido además de mostrar el movimiento del objeto tangible físico.

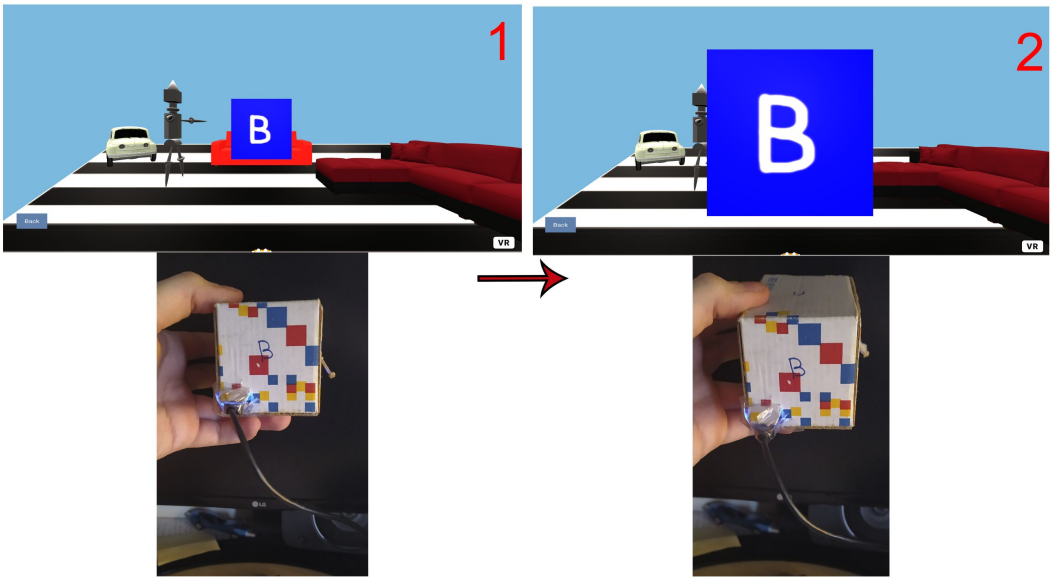


Figura 6.35: Mover posteriormente despacio en Mover

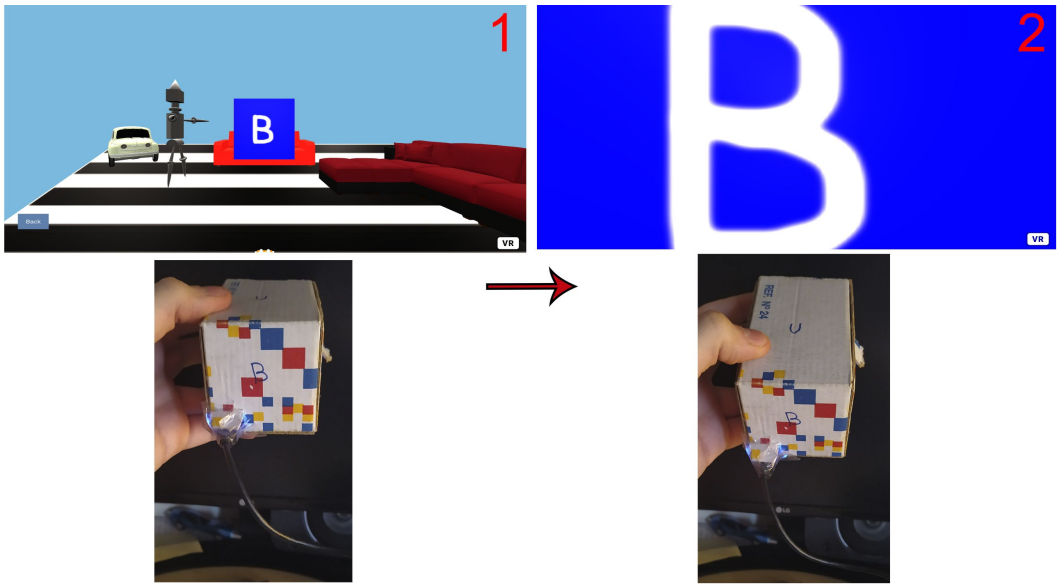
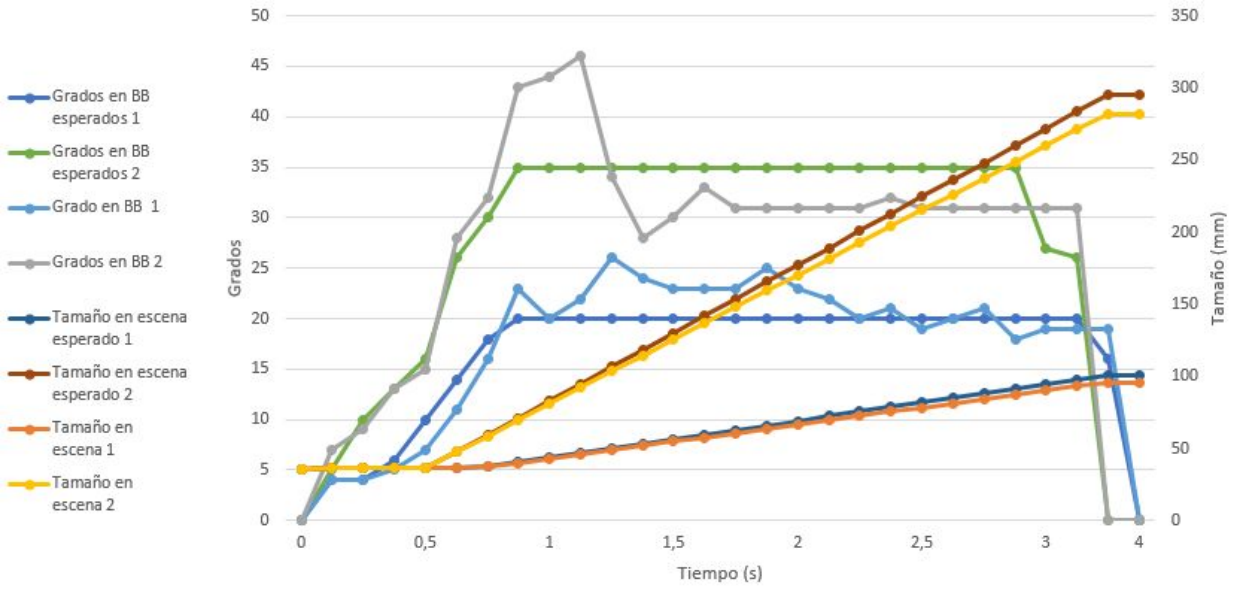


Figura 6.36: Mover posteriormente rápido en Mover

Para comprobar que ocurre lo que se espera, que cuando se rota con un ángulo pequeño, entre 15º y 26º, se mueve más lentamente y cuando es mayor, entre 27º y 55º, la velocidad es mayor, se utiliza la gráfica 6.37 con ambos movimientos en un rango de 4 segundos.

6.37: Gráfica del tamaño del objeto tangible al desplazarse posteriormente en Mover



Las series indican los grados que se obtienen del eje X del acelerómetro de la BB al girarla y el tamaño es el tamaño del modelo del objeto tangible que se ve en la pantalla.

Para la inclinación pequeña se representa con el nombre de la serie y un 1 y para la inclinación alta con el nombre de la serie y un 2. Con la inclinación pequeña, los grados obtenidos de la BB se encuentran en torno a los esperados, mientras que el tamaño aumenta de forma muy similar al esperado. Con la inclinación más alta, se obtiene que los grados obtenidos se encuentran alrededor de los esperados y el tamaño del modelo aumenta de forma muy similar al esperado. El tamaño se mide en milímetros en la pantalla física del ordenador, resultando en lo que se ve en las figuras 6.35 y 6.36 junto a la BB.

La inclinación de las series referentes al tamaño es igual a las series esperadas y los grados obtenidos están cerca de las series esperadas.

4) Mover hacia arriba

Basado en el caso de uso 3.21. En esta experimentación se hace uso de la BeagleBone, por lo que es necesario calcular el retardo entre los datos que da y los que se muestran en Ubuntu. En la tabla E.7 se muestra un retraso de 9 milisegundos.

Se busca comprobar que cuando el objeto tangible tiene una rotación mayor a cierto rango hacía atrás, el objeto tangible se desplaza verticalmente hacia arriba. En la figura 6.38 se ve la rotación del objeto tangible y su representación rotando de igual forma.

Para comprobar que ocurre lo que se espera, que cuando se rota más de 55º el objeto tangible se mueve verticalmente hacia arriba, se utiliza la gráfica 6.39 de una muestra de 3,5 segundos.

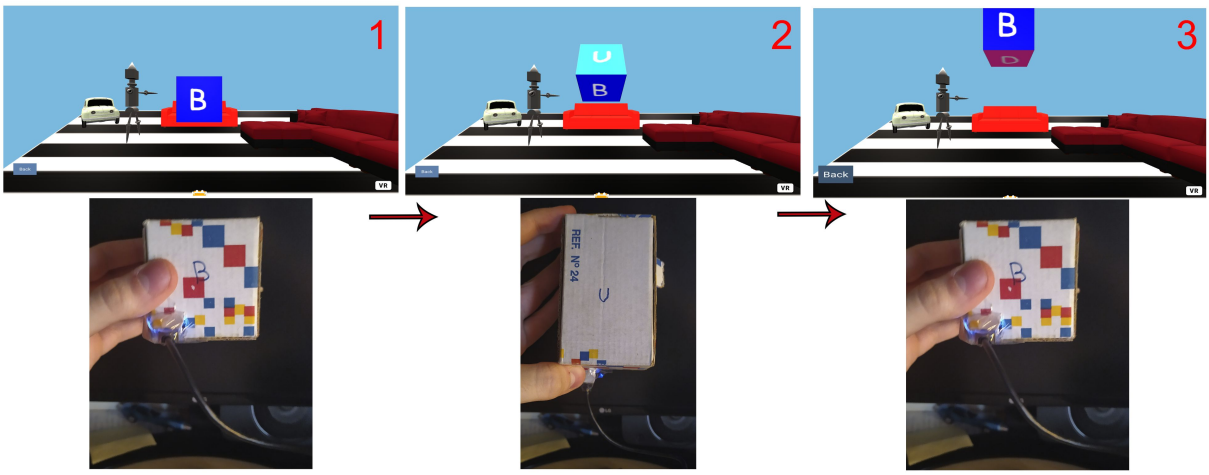
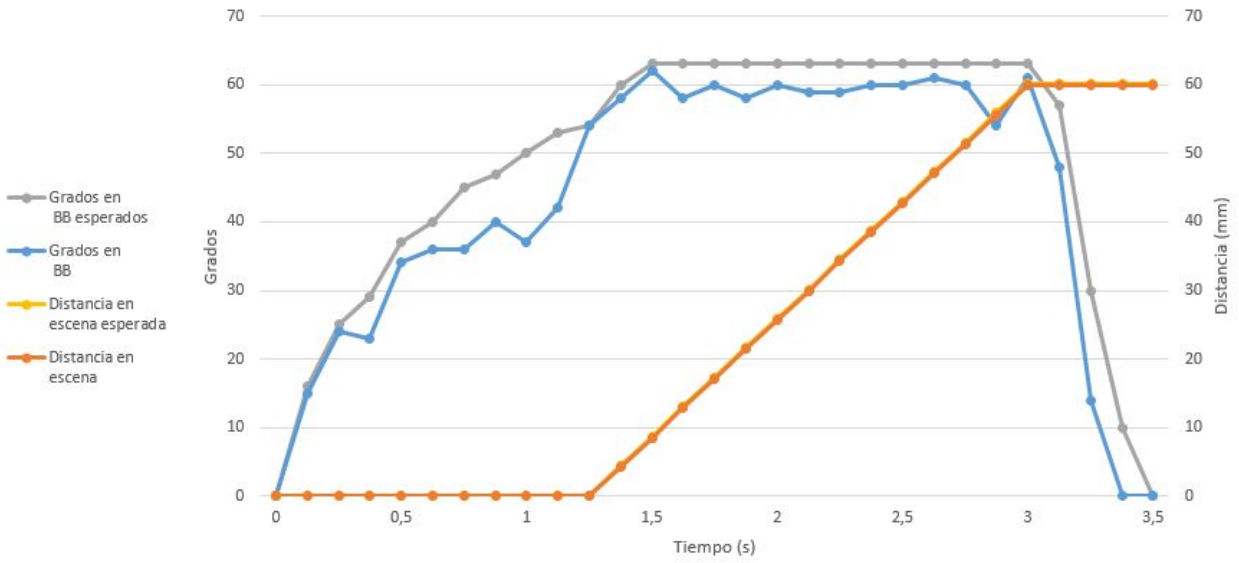


Figura 6.38: Rotar hacia atrás el objeto tangible en Mover

Las series indican los grados que se obtienen del eje X del acelerómetro de la BB al girarla y la distancia es la distancia vertical en pantalla que el modelo del objeto tangible recorren en esa muestra.

6.39: Gráfica de la distancia recorrida cuando se rota hacia atrás en Mover



Cuando se inclina la BB por encima de los 55º, como se ve en la segunda imagen de la figura 6.38, la distancia empieza a aumentar. En la gráfica se puede ver la serie de

los grados de la BB es similar a la esperada y que la distancia en escena prácticamente igual a la esperada.

La serie de la distancia tienen la misma inclinación y comienzan y terminan a la vez que la distancia esperada, mientras que la serie de los grados es muy similar a la esperada.

5) Mover hacia abajo

Basado en el caso de uso 3.22. En esta experimentación se hace uso de la BeagleBone, por lo que es necesario calcular el retardo entre los datos que da y los que se muestran en Ubuntu. En la tabla E.8 se muestra un retraso de 9 milisegundos.

Se busca comprobar que cuando el objeto tangible tiene una rotación mayor a cierto rango hacía adelante, el objeto tangible se desplaza verticalmente hacia abajo. En la figura 6.40 se ve la rotación del objeto tangible y su representación rotando de igual forma.

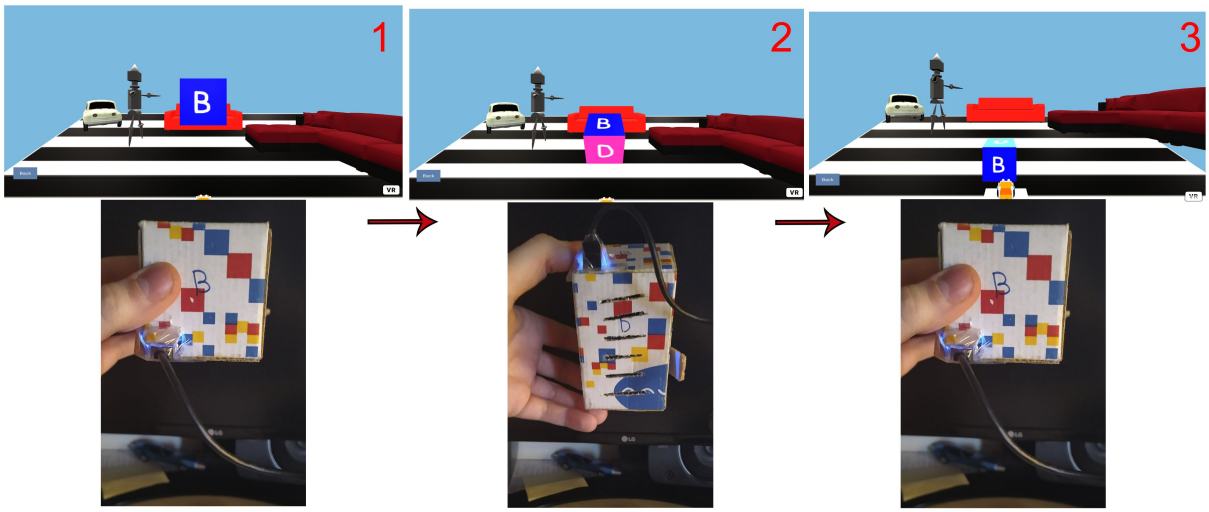
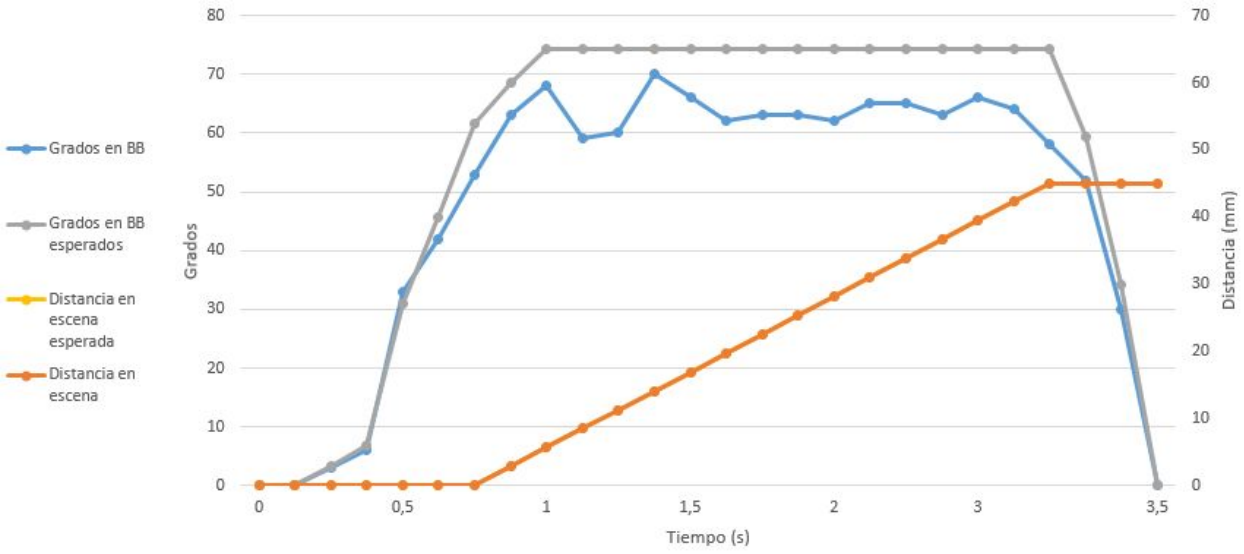


Figura 6.40: Rotar hacia adelante el objeto tangible en Mover

Para comprobar que ocurre lo que se espera, que cuando se rota más de 55º el

objeto tangible se mueve verticalmente hacia arriba, se utiliza la gráfica 6.41 de una muestra de 3,5 segundos.

6.41: Gráfica de la distancia recorrida cuando se rota hacia adelante en Mover



Las series indican los grados que se obtienen del eje X del acelerómetro de la BB al girarla y la distancia es la distancia vertical en pantalla que el modelo del objeto tangible recorren en esa muestra.

Cuando se inclina la BB por encima de los 55º, como se ve en la segunda imagen de la figura 6.40, la distancia empieza a aumentar. En la gráfica se puede ver la serie de los grados de la BB es similar a la esperada y que la distancia en escena prácticamente igual a la esperada.

La serie de la distancia tienen la misma inclinación y comienzan y terminan a la vez que la distancia esperada, mientras que la serie de los grados es muy similar a la esperada.

6) Interactuar con objeto

Basado en el caso de uso 3.23. En esta experimentación se hace uso de la BeagleBone, pero lo que se busca en esta no es comparar el movimiento del objeto tangible y su representación, sino mostrar las interacciones de la representación con los objetos de la escena. Esta experimentación se divide en dos parte para comprobar que ocurre que el sofá de la derecha en la figura 6.42 muestre un mensaje diciendo lo que es, y que el robot de la figura 6.43 muestre un carte y se retire.

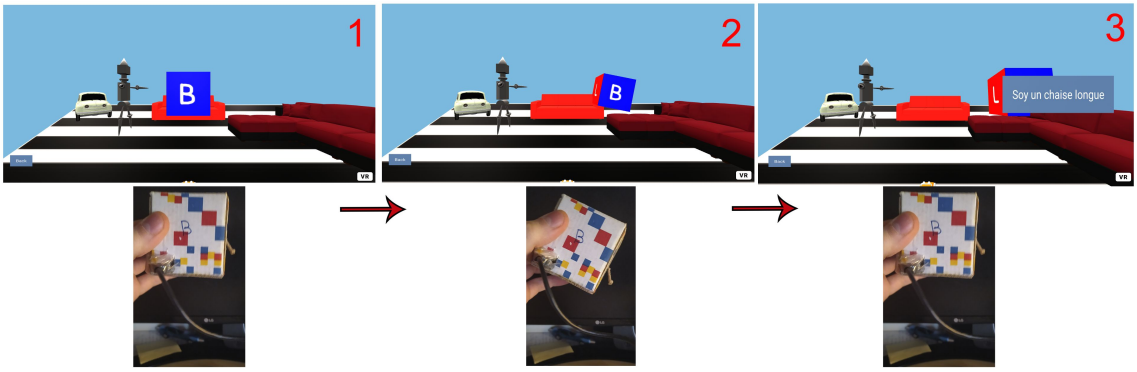


Figura 6.42: Interacción con el modelo del chaise longue en Mover

Rotando el objeto físico, se puede hacer moverse a la representación hacia donde se quiera. En esta ocasión, después de desplazarlo por las pantallas *1* y *2*, en la tercera se toca el sofá con la representación y el primero muestra un cartel diciendo que no es un sofá, sino un 'chaise longue'.

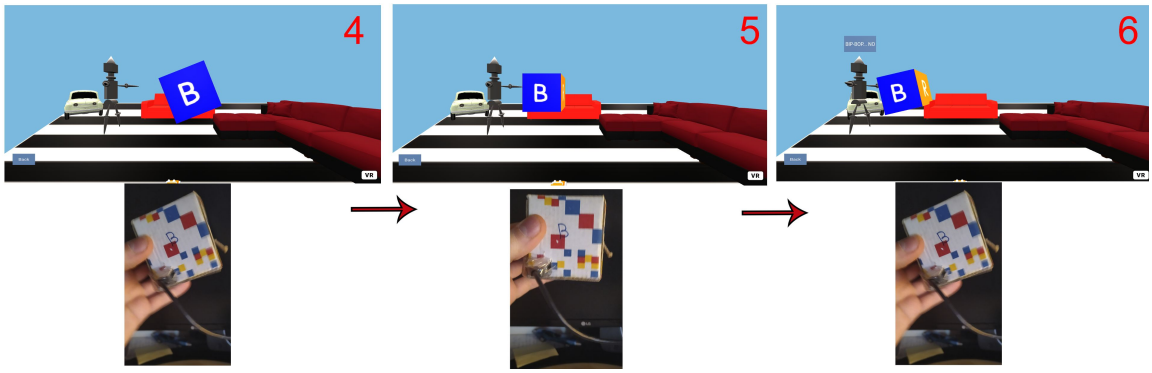


Figura 6.43: Interacción con el modelo del robot en Mover

Haciendo lo mismo que en el anterior caso, se lleva el objeto hasta el robot de la escena y éste, en lugar de decir que es un robot, se retira un poco y muestra un cartel con el texto 'BIP-BOP... NO'.

7) Salir de mover

Basado en el caso de uso 3.24. Desde la escena de Mover como se muestra en la pantalla *1* de la figura 6.44, el usuario puede volver al menú principal. Para ello, al poner el ratón sobre el botón *'Back'* como en la pantalla *2*, éste botón se hará grande. Al pulsarlo, hará una animación y pasará a la escena del menú principal en la pantalla *3*, haciendo la animación inversa que cuando se entró en la escena Mover, ocupando todo como en la pantalla *3* hasta llegar al menú principal en la pantalla *5*, pasando por la pantalla *4*.

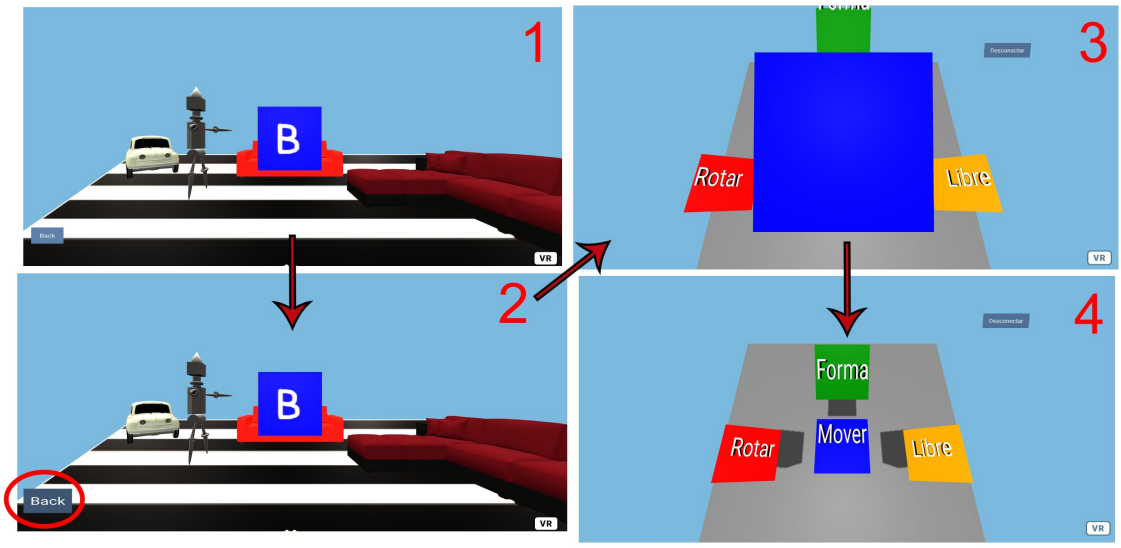


Figura 6.44: Salir de Mover

6.6. Libre

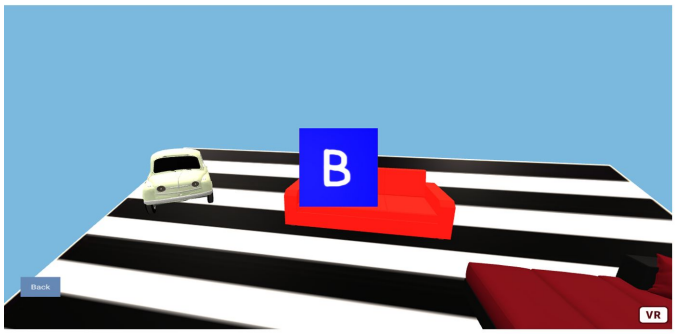


Figura 6.45: Escena de Libre

En esta escena se muestra el objeto que representa el objeto tangible con la forma inicial o con la seleccionada en la escena 6.3, un botón en la esquina inferior izquierda en el que pone 'Back' y una escena con diversos objetos en el escenario.

- **Objeto tangible:** la representación del objeto tangible de HITO. En esta escena rotará levemente pero estará enfocado en desplazarse por toda la escena, moviéndose hacia la izquierda cuando el objeto físico rota hacia la izquierda y viceversa. También pasa con las rotaciones posteriores y frontales que acercará y alejará el objeto del punto de vista. Si se inclina mucho posterior o frontalmente, el objeto ascenderá o descenderá respectivamente. En esta ocasión la cámara se moverá como detrás del objeto tangible como si de un avión se tratase. Al desplazarse junto al objeto, da la sensación de exploración en el entorno tridimensional, pudiendo ayudar a entender estos entornos.

- **Objetos del escenario:** los objetos de la escena no son interactuables, pero se pueden atravesar mientras se explora la escena.

- **Back:** al poner el ratón sobre él resaltará y al pulsarlo llevará a la escena 6.2 del menú principal.

Esta escena consta de las siguientes experimentaciones basadas en los casos de uso:

1) Desplazarse frontalmente

Basado en el caso de uso 3.26. En esta experimentación se hace uso de la Beagle-Bone, por lo que es necesario calcular el retardo entre los datos que da y los que se muestran en Ubuntu. En la tabla E.9 se muestra un retraso de 16 milisegundos.

Se busca comprobar que cuando el objeto tangible tiene una rotación pequeña hacia adelante, el desplazamiento ocurre lentamente y que, cuando la rotación es mayor a cierto ángulo, el desplazamiento frontal es más rápido. En la figura 6.46 se muestra como sería un desplazamiento lento y en la figura 6.47 uno rápido además de mostrar el movimiento del objeto tangible físico.

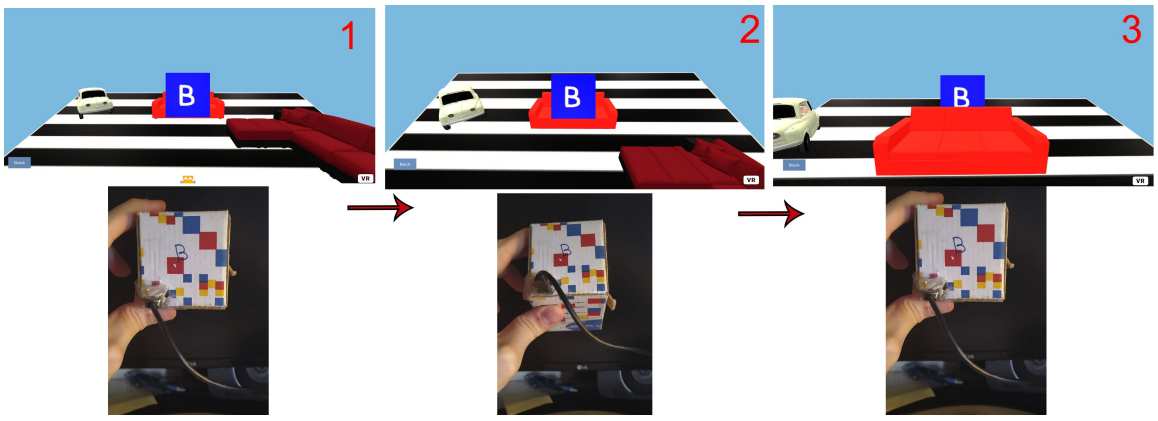


Figura 6.46: Desplazar frontalmente despacio en Libre

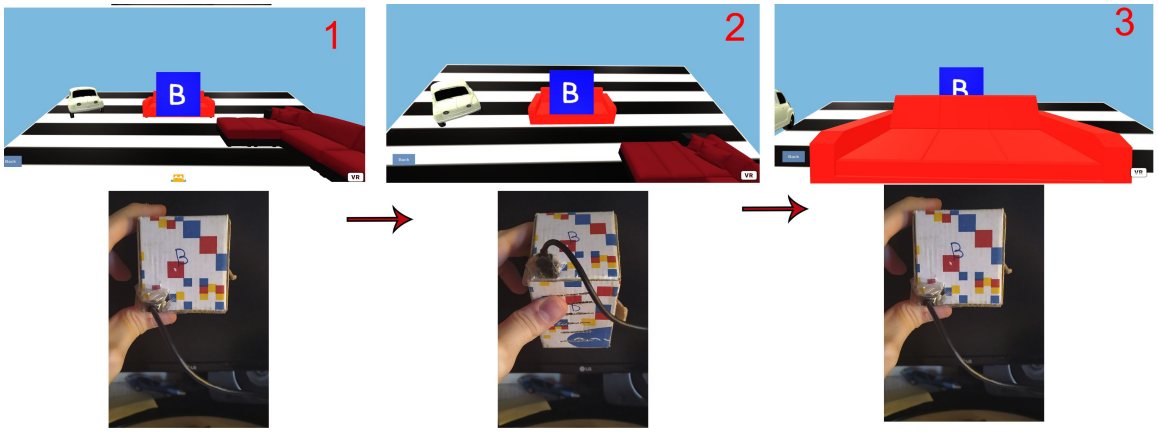
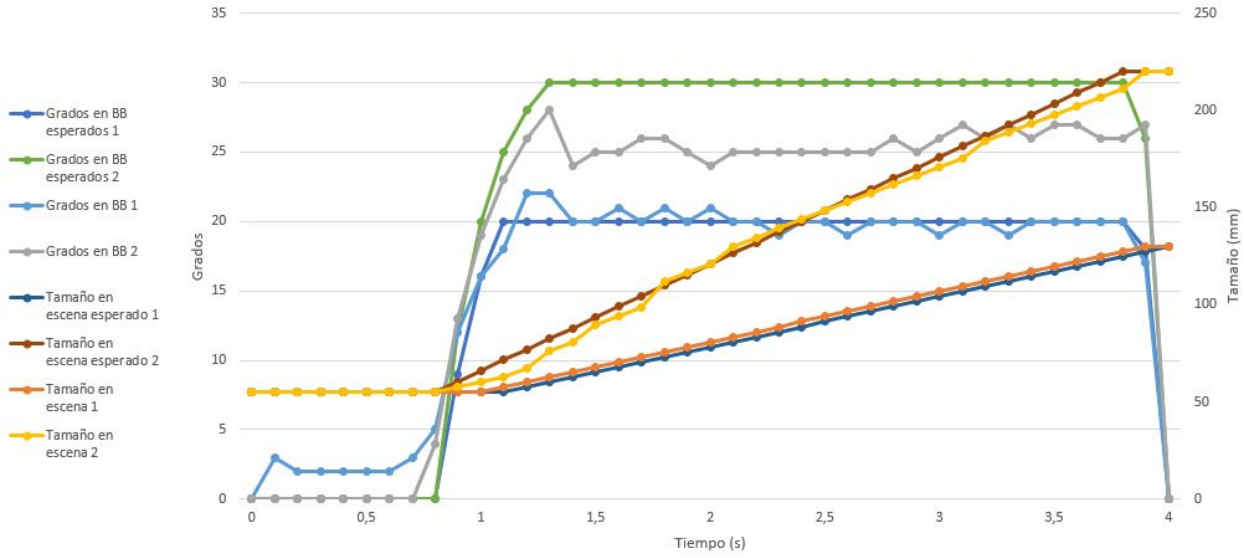


Figura 6.47: Desplazar frontalmente rápido en Libre

Para comprobar que ocurre lo que se espera, que cuando se rota con un ángulo pequeño, entre 15º y 26º, se mueve más lentamente y cuando es mayor, entre 27º y 55º, la velocidad es mayor, se utiliza la gráfica 6.48 con ambos movimientos en un rango de 4 segundos.

Las series indican los grados que se obtienen del eje X del acelerómetro de la BB al girarla y el tamaño es el tamaño del ancho del sillón rojo que se ve en la pantalla, porque no se puede medir el tamaño del modelo de la BB ya que la cámara sigue al objeto.

6.48: Gráfica del tamaño del sofá rojo al desplazarse el objeto frontalmente en Libre



Para la inclinación pequeña se representa con el nombre de la serie y un 1 y para la inclinación alta con el nombre de la serie y un 2. Los grados obtenidos de la BB se encuentran en torno a los esperados, mientras que el tamaño aumenta de forma muy similar al esperado. Con la inclinación más alta, se obtienen grados un poco por debajo de los esperados, pero en el rango mencionado, y el tamaño aumenta de forma muy similar al esperado. El tamaño es medido en milímetros en la pantalla física del ordenador, resultando en lo que se ve en las figuras 6.46 y 6.47 junto a la BB.

La inclinación de las series de los tamaños es prácticamente igual a la de las series esperada, mientras que las series de los grados se encuentran en torno a las esperadas.

2) Desplazarse posteriormente

Basado en el caso de uso 3.27. En esta experimentación se hace uso de la BeagleBone, por lo que es necesario calcular el retardo entre los datos que da y los que se muestran en Ubuntu. En la tabla E.10 se muestra un retraso de 16 milisegundos.

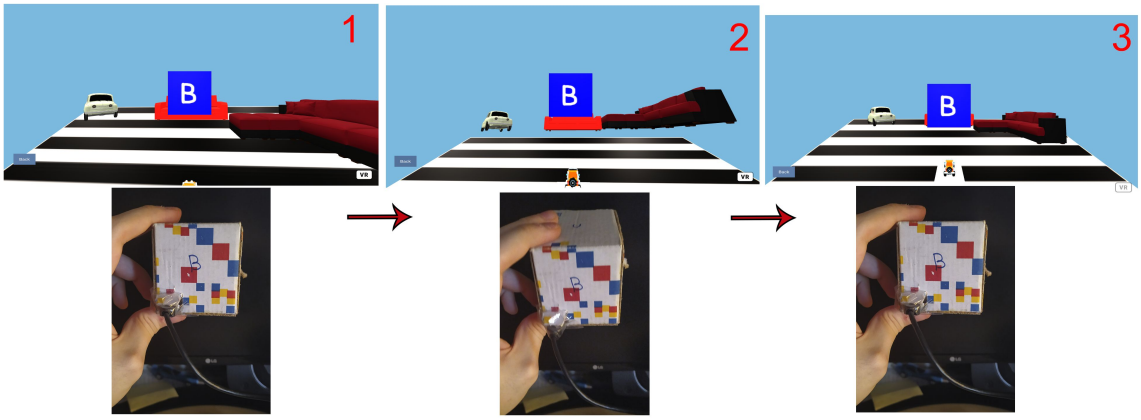


Figura 6.49: Desplazar posteriormente despacio en Libre

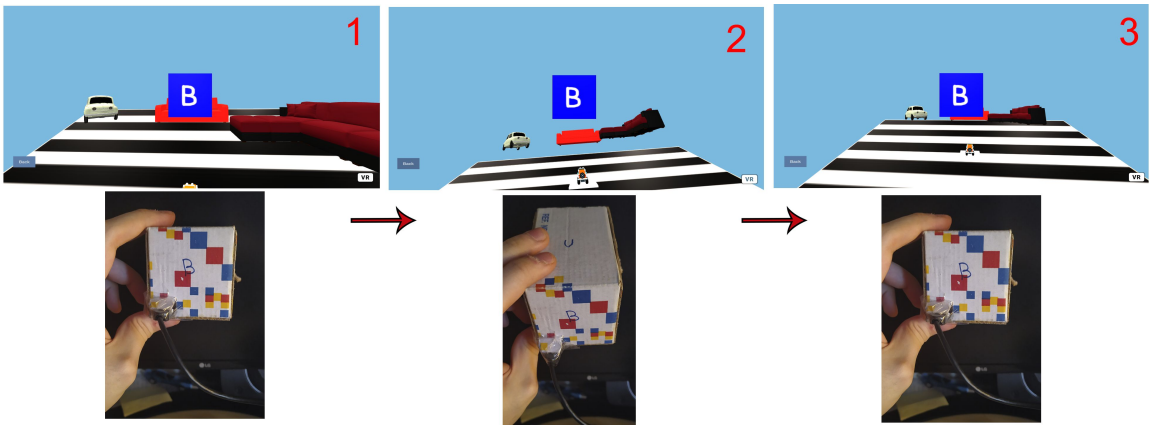


Figura 6.50: Desplazar posteriormente rápido en Libre

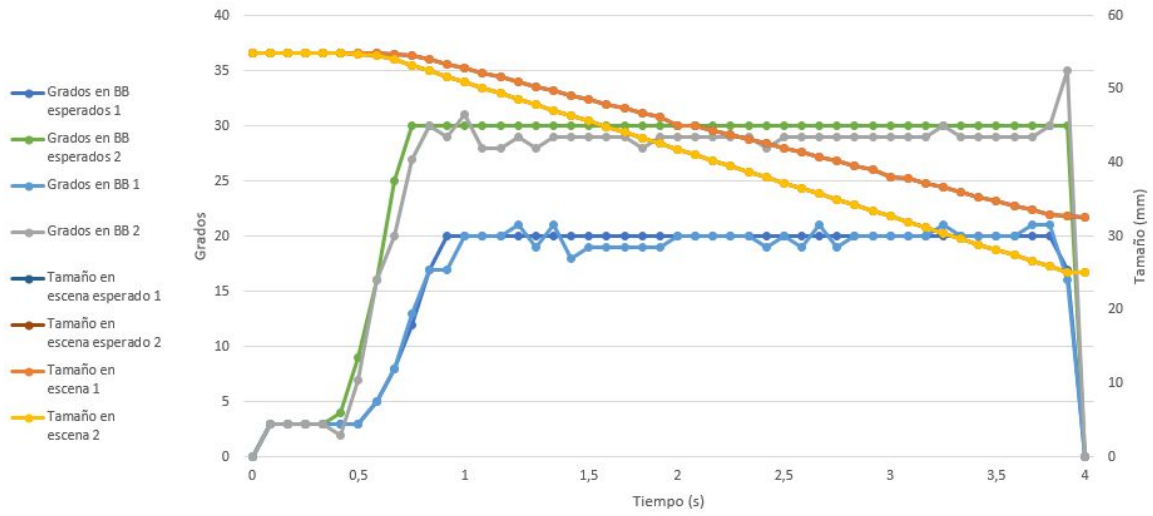
Se busca comprobar que cuando el objeto tangible tiene una rotación pequeña hacia atrás, el desplazamiento ocurre lentamente y que, cuando la rotación es mayor

a cierto ángulo, el desplazamiento frontal es más rápido. En la figura 6.49 se muestra como sería un desplazamiento lento y en la figura 6.50 uno rápido además de mostrar el movimiento del objeto tangible físico.

Para comprobar que ocurre lo que se espera, que cuando se rota con un ángulo pequeño, entre 15º y 26º, se mueve más lentamente y cuando es mayor, entre 27º y 55º, la velocidad es mayor, se utiliza la gráfica 6.51 con ambos movimientos en un rango de 4 segundos.

Las series indican los grados que se obtienen del eje X del acelerómetro de la BB al girarla y el tamaño es el tamaño del ancho del sillón rojo que se ve en la pantalla, porque no se puede medir el tamaño del modelo de la BB ya que la cámara sigue al objeto.

6.51: Gráfica del tamaño del sofá rojo al desplazarse el objeto posteriormente en Libre



Para la inclinación pequeña se representa con el nombre de la serie y un 1 y para la inclinación alta con el nombre de la serie y un 2. Los grados obtenidos de la BB se encuentran en torno a los esperados, mientras que el tamaño disminuye de forma muy similar al esperado. Con la inclinación más alta, se obtienen grados un poco por

debajo de los esperados, pero en el rango mencionado, y el tamaño disminuye de forma muy similar al esperado. El tamaño es medido en milímetros en la pantalla física del ordenador, resultando en lo que se ve en las figuras 6.46 y 6.47 junto a la BB.

La inclinación de las series de los tamaños es prácticamente igual a la de las series esperada, mientras que las series de los grados se encuentran en torno a las esperadas.

3) Rotar horizontalmente

Basado en el caso de uso 3.28. En esta experimentación se hace uso de la BeagleBone, por lo que es necesario calcular el retardo entre los datos que da y los que se muestran en Ubuntu. En la tabla E.11 se muestra un retraso de 19 milisegundos.

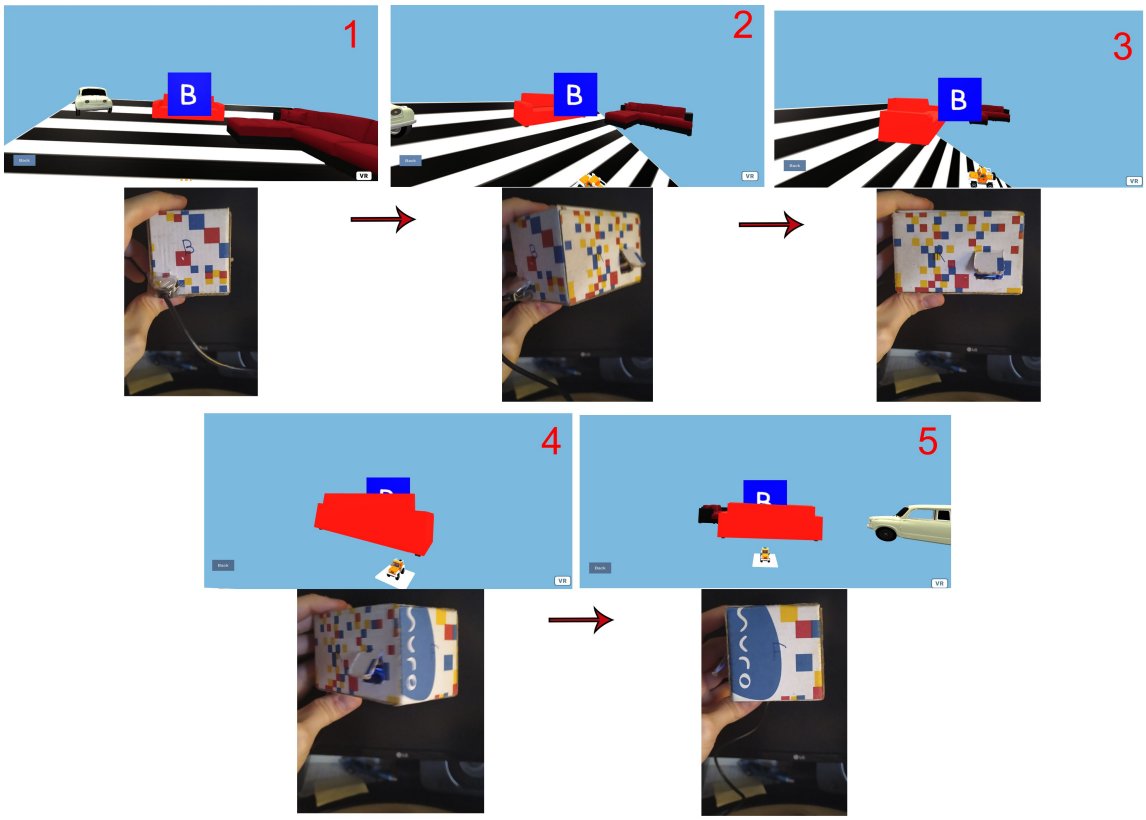


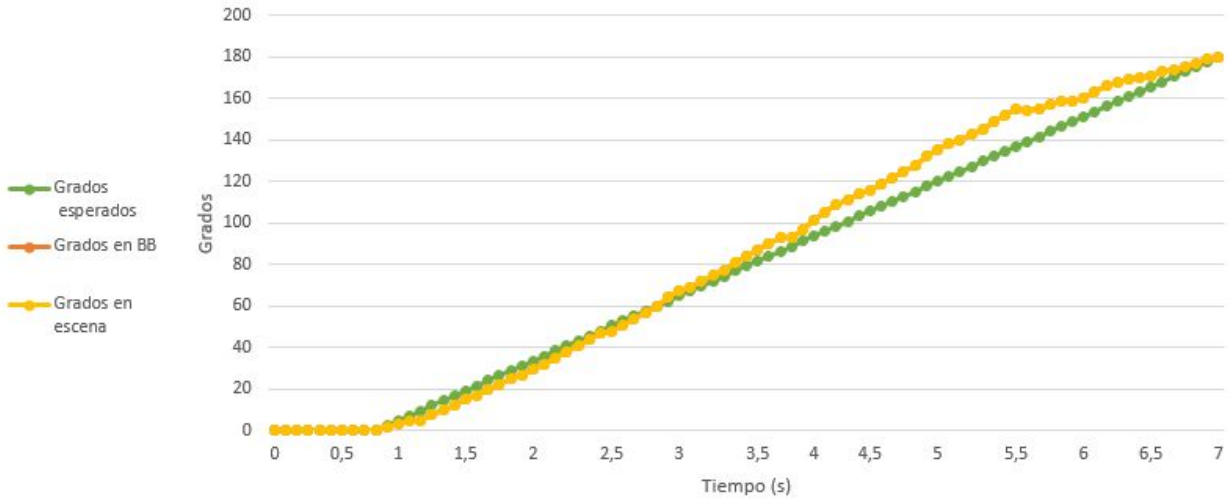
Figura 6.52: Rotación horizontal del objeto tangible e imitación en la interfaz en Libre

La rotación mostrada en la figura 6.52 donde se ve que al rotar hacia la derecha

horizontalmente el objeto tangible, su representación en la interfaz sigue su movimiento a lo largo de las 5 pantallas mostradas. Para comprobar que la rotación es de 180 grados sobre el eje horizontal y que gira a una velocidad uniforme durante 7 segundos como se espera, se muestra la gráfica 6.53.

En la gráfica se muestran las series de datos de los grados que tanto la BeagleBone como el objeto en escena muestran utilizando los valores del eje Z del giroscopio y los valores obtenidos del eje Z del modelo del objeto tangible respectivamente. También se encuentra la serie de los grados esperados para comparar con las otras dos. El movimiento que representa la gráfica es el de la figura 6.52.

6.53: Gráfica de rotación horizontal uniforme en el tiempo en Libre



6.21: Gráfica de rotación horizontal uniforme en el tiempo en RotarLas series de la BB y de la escena se solapan al dar ambos el mismo resultado, queresulta estar un poco por encima de los valores esperados. Se observa que al mover el objeto tangible a una velocidad uniforme, el aumento de los grados es uniforme también. El objeto tangible comienza a rotar a los 0.5 segundos, por lo que los datos anteriores corresponden a pantalla *1*. Alrededor de 2 segundos después, se alcanzan los 45º como se muestra en la pantalla *2* y, llegados a los 4 segundos, se alcanzan los 90º y la pantalla *3*. Siguiendo

el movimiento, a los 5 segundos más o menos, se alcanzan los 135º que se muestran en la pantalla *4* y termina con el giro de 180º con la pantalla *5* a los 7 segundos. Cuando se alcanza la cuarta imagen, el objeto que hace de suelo en la escena deja de verse porque la cámara se mueve con el objeto tangible.

Las series tienen una inclinación similar y varían al mismo ritmo que la serie de grados esperados.

4) Mover lateralmente

Basado en el caso de uso 3.29. En esta experimentación se hace uso de la Beagle-Bone, por lo que es necesario calcular el retardo entre los datos que da y los que se muestran en Ubuntu. En la tabla E.12 se muestra un retraso de 26 milisegundos.

Se busca comprobar que cuando el objeto tangible tiene una rotación pequeña sobre sí mismo hacia la izquierda o hacia la derecha, el desplazamiento ocurre lentamente y que, cuando la rotación es mayor a cierto ángulo, el desplazamiento lateral es más rápido. El objeto físico solo rota, no se desplaza. En la figura 6.54 se muestra como sería un desplazamiento lento y en la figura 6.55 uno rápido además de mostrar el movimiento del objeto tangible físico.

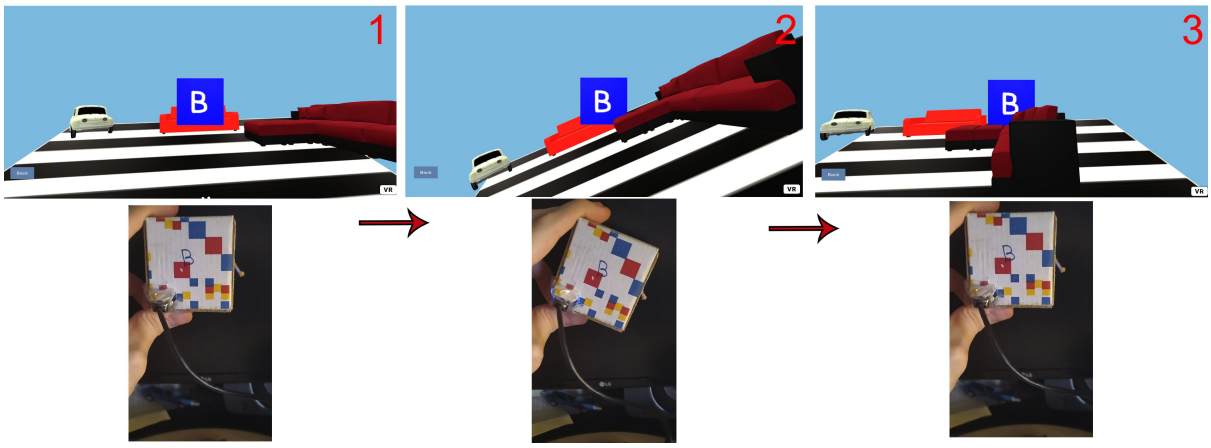


Figura 6.54: Mover lateralmente despacio en Libre

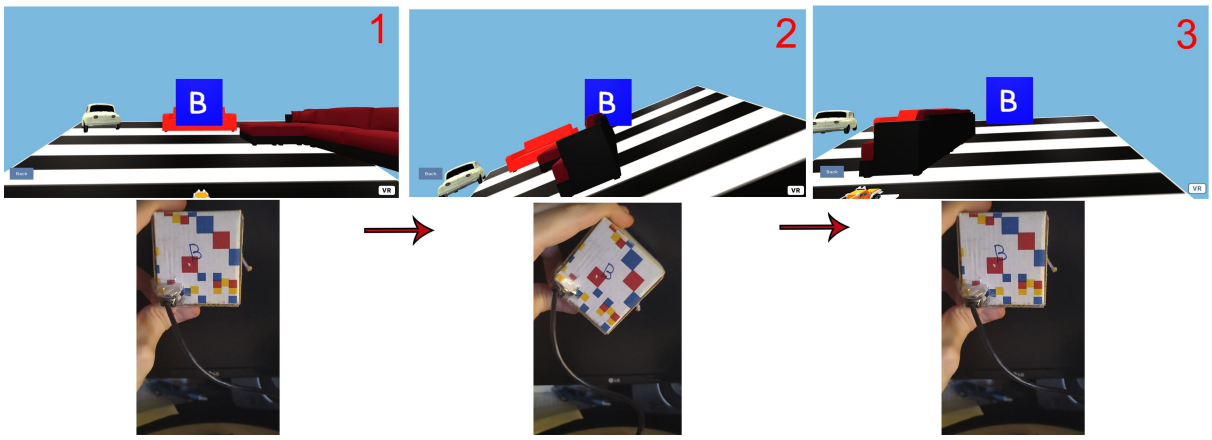
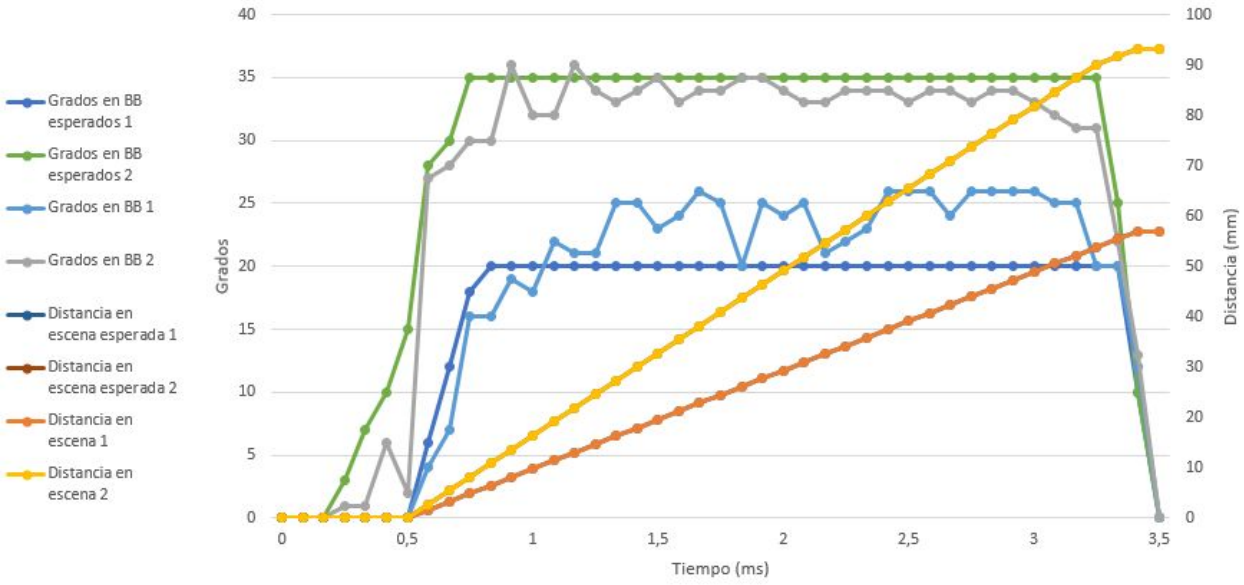


Figura 6.55: Mover lateralmente rápido en Libre

Para comprobar que ocurre lo que se espera, que cuando se rota con un ángulo pequeño, entre 15º y 26º, se mueve más lentamente y cuando es mayor, entre 27º y 55º, la velocidad es mayor, se utiliza la gráfica 6.56 con ambos movimientos en un rango de 3,5 segundos.

6.56: Gráfica del desplazamiento lateral en Libre



Las series indican los grados que se obtienen del eje Y de la BB del acelerómetro al girarla y la distancia recorrida por el modelo en pantalla. De ambos valores se

muestran los esperados.

Cuando se realiza una inclinación pequeña, representado este movimiento con el nombre de la serie y un 1, para un desplazamiento más lento del objeto de la escena,los grados obtenidos se mueven en torno a los esperados, mientras que la distancia es muy similar a la esperada. La inclinación más alta se representa con el nombre dela serie y un 2 y se obtiene que los grados obtenidos se encuentran alrededor de los esperados y la distancia recorrida es muy similar a la esperada. La distancia es medida en milímetros en la pantalla física del ordenador, resultando en lo que se ve en la figura 6.57, siendo el 1 de esta figura la posición inicial, el 2 cuando el desplazamiento es lento y el 3 cuando es rápido. En la gráfica se muestra que cuanto más grados, mayor es el desplazamiento, tal y como se esperaba.

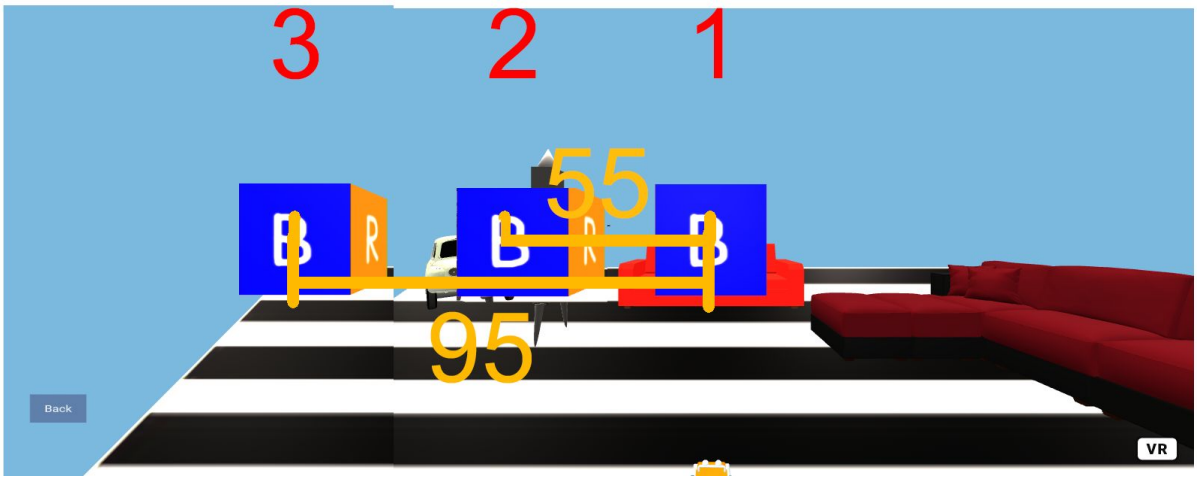


Figura 6.57: Distancia recorrida en el desplazamiento lateral en Libre

La inclinación de las series que se refieren a la distancia tienen la misma inclinación e inician y terminan en el mismo punto cuando los grados disminuyen de los valores mencionados anteriormente. Las series de los grados se ajustan se mueven en torno a los valores esperados.

5) Mover hacia arriba

Basado en el caso de uso 3.30. En esta experimentación se hace uso de la Beagle-Bone, por lo que es necesario calcular el retardo entre los datos que da y los que se muestran en Ubuntu. En la tabla E.13 se muestra un retraso de 19 milisegundos.

Se busca comprobar que cuando el objeto tangible tiene una rotación mayor a cierto rango hacía atrás, el objeto tangible se desplaza verticalmente hacia arriba. En la figura 6.58 se ve la rotación del objeto tangible y su representación rotando de igual forma.

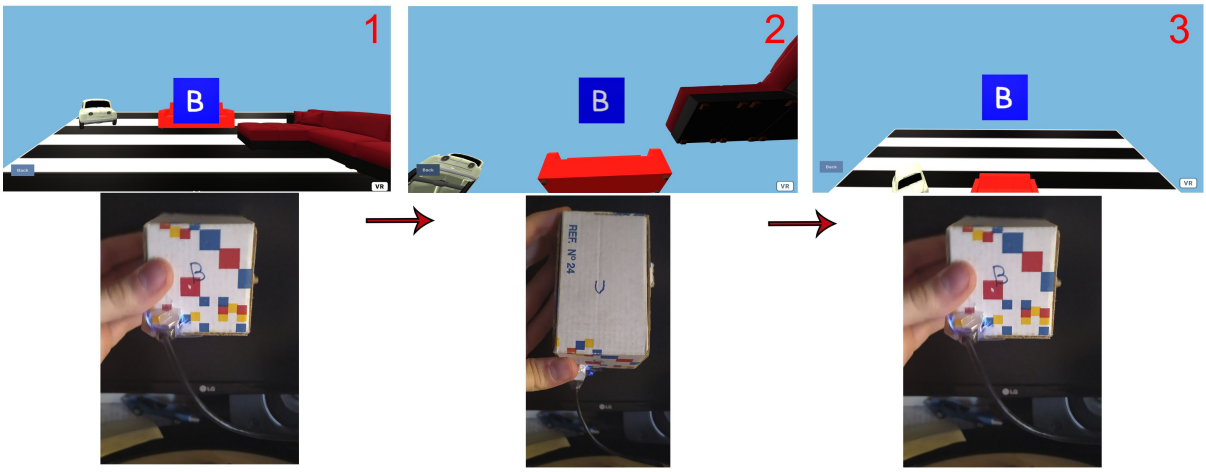
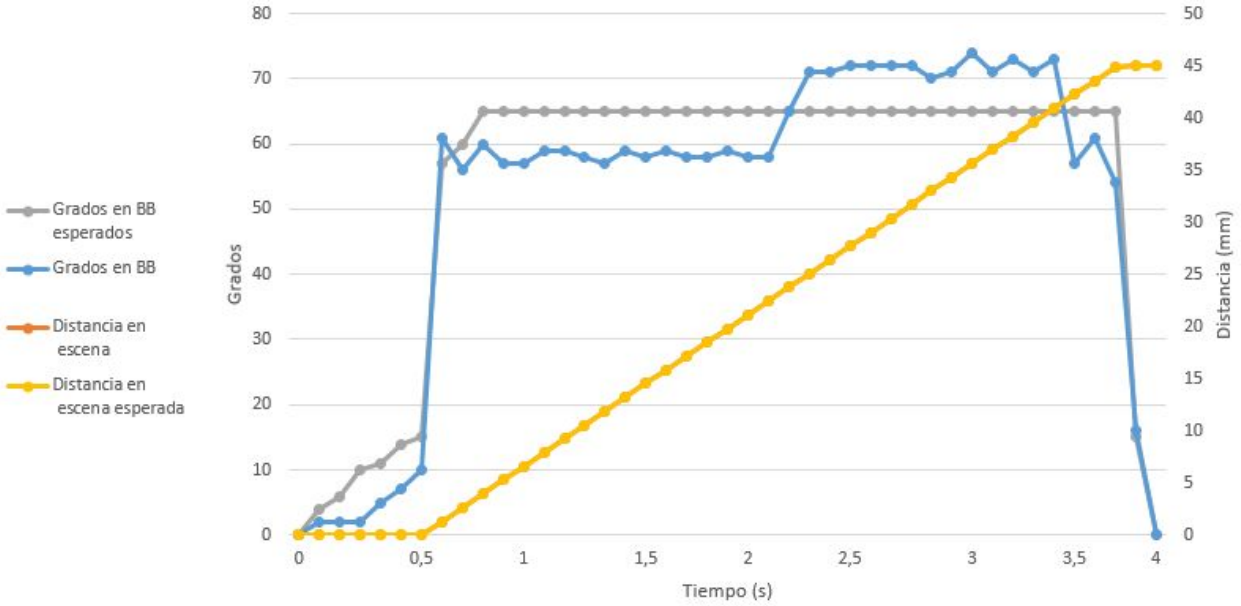


Figura 6.58: Rotar hacia atrás el objeto tangible en Libre

Para comprobar que ocurre lo que se espera, que cuando se rota más de 55º el objeto tangible se mueve verticalmente hacia arriba, se utiliza la gráfica 6.59 de una muestra de 4 segundos.

Las series indican los grados que se obtienen del eje X del acelerómetro de la BB al girarla y la distancia es la distancia vertical en milímetros en pantalla que el modelo del objeto tangible recorren en esa muestra.

6.59: Gráfica de la distancia recorrida cuando se rota hacia atrás en Libre



Cuando se inclina la BB por encima de los 55º, como se ve en la segunda imagen dela figura 6.58, la distancia empieza a aumentar. En la gráfica se puede ver la serie de los grados de la BB es similar a la esperada y que la distancia en escena prácticamente igual a la esperada.

La serie de la distancia tienen la misma inclinación y comienzan y terminan a la vez que la distancia esperada, mientras que la serie de los grados es muy similar a la esperada.

6) Mover hacia abajo

Basado en el caso de uso 3.31. En esta experimentación se hace uso de la BeagleBone, por lo que es necesario calcular el retardo entre los datos que da y los que se muestran en Ubuntu. En la tabla E.14 se muestra un retraso de 36,93 milisegundos.

Se busca comprobar que cuando el objeto tangible tiene una rotación mayor a cierto rango hacía adelante, el objeto tangible se desplaza verticalmente hacia abajo.

En la figura 6.60 se ve la rotación del objeto tangible y su representación rotando de igual forma.

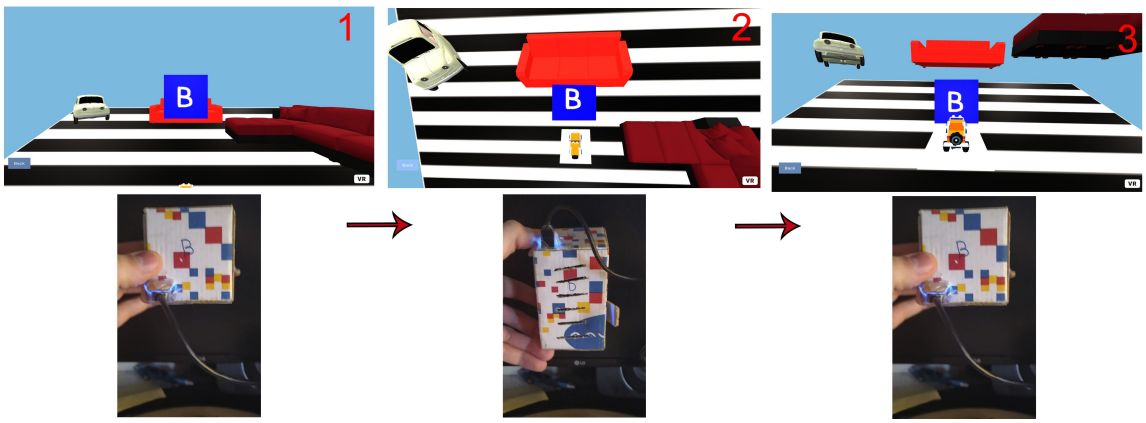
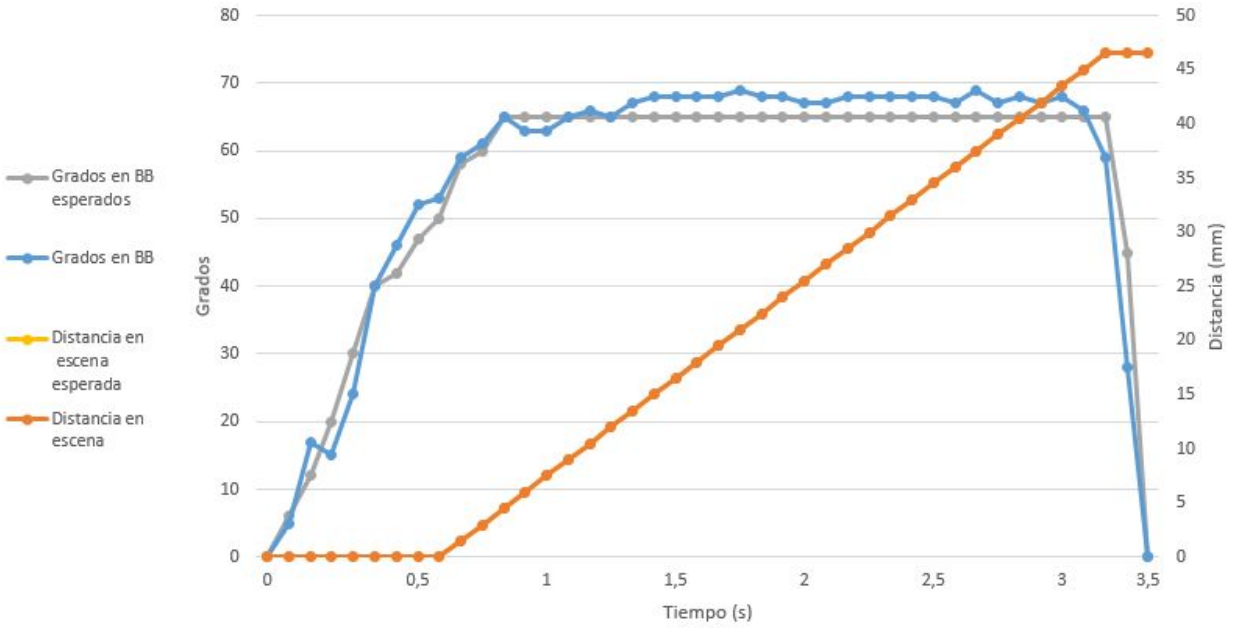


Figura 6.60: Rotar hacia adelante el objeto tangible en Libre

Para comprobar que ocurre lo que se espera, que cuando se rota más de 55º el objeto tangible se mueve verticalmente hacia abajo, se utiliza la gráfica 6.60 de una muestra de 3,5 segundos.

6.61: Gráfica de la distancia recorrida cuando se rota hacia adelante en Libre



Las series indican los grados que se obtienen del eje X del acelerómetro de la BB al girarla y la distancia es la distancia vertical en milímetros en pantalla que el modelo del objeto tangible recorren en esa muestra.

Cuando se inclina la BB por encima de los 55º, como se ve en la segunda imagen dela figura 6.60, la distancia empieza a aumentar. En la gráfica se puede ver la serie de los grados de la BB es similar a la esperada y que la distancia en escena prácticamente igual a la esperada.

La serie de la distancia tienen la misma inclinación y comienzan y terminan a la vez que la distancia esperada, mientras que la serie de los grados es muy similar a la esperada.

7) Mover en diagonal

Basado en el caso de uso 3.32. En esta experimentación se hace uso de la BeagleBone, por lo que es necesario calcular el retardo entre los datos que da y los que se muestran en Ubuntu. En la tabla E.15 se muestra un retraso de 32,67 milisegundos.

Se busca comprobar que cuando el objeto tangible rota en un poco sobre sí mismo, un poco en horizontal y otro poco en vertical, el resultado es un desplazamiento en diagonal. Para ello, se han tomado muestras de los ejes X e Y del acelerómetro y del eje Z del giroscopio y se han puesto en comparación con lo que se muestra en la figura 6.62 utilizando la gráfica 6.63 junto con los valores esperados.

En la primera pantalla de la figura 6.62 el objeto está en reposo como al inicio de la gráfica. En la segunda pantalla está en movimiento y todos los ejes angulados, lo que se muestra entre los segundos 0.5 y 4 de la gráfica. En la tercera pantalla el objeto vuelve a estar en reposo y en la gráfica los ejes vuelven a dar 0 como resultado. Se puede ver en la imagen cómo cambia de posición el objeto tangible entre la primera y la última pantalla.

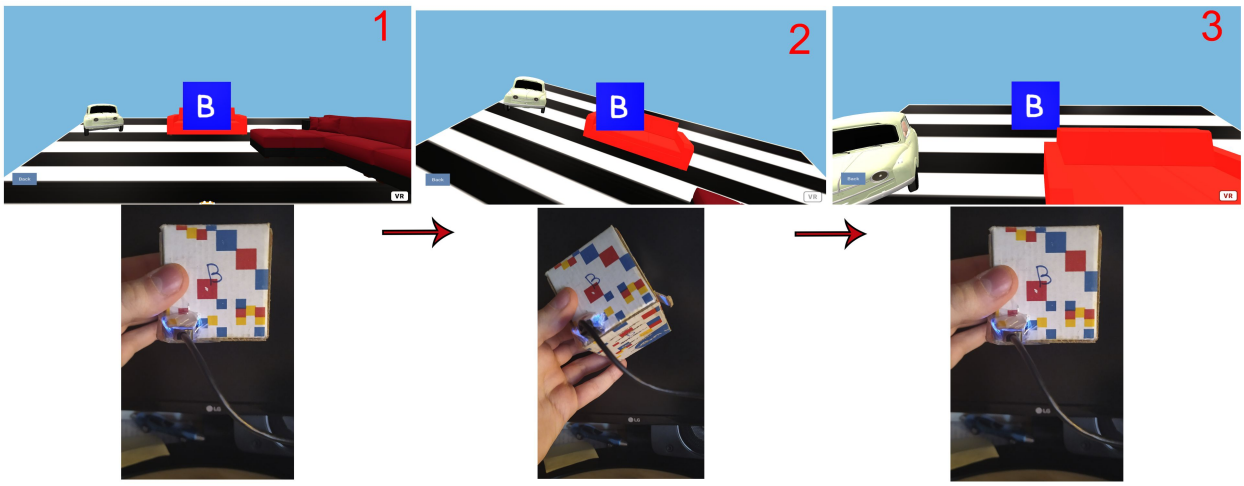
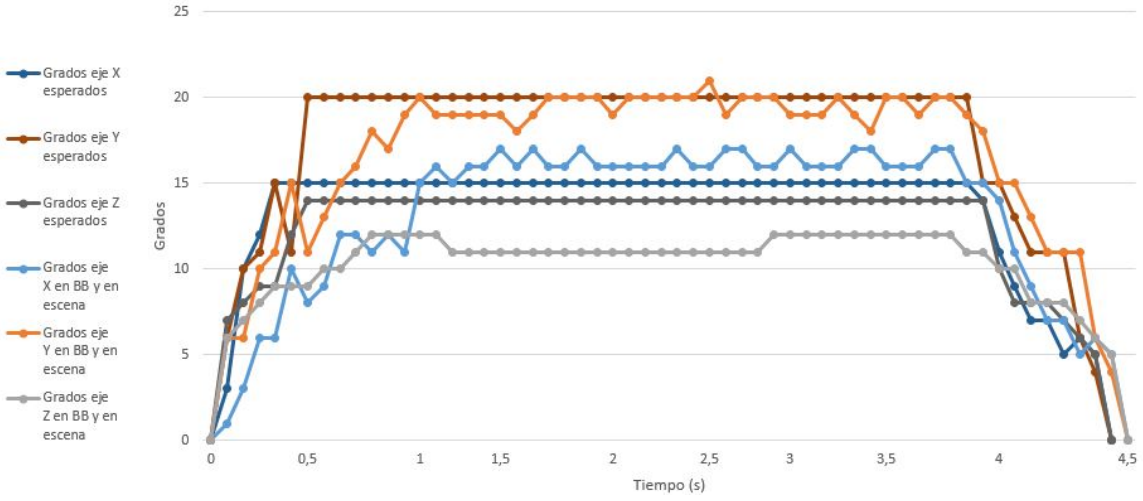


Figura 6.62: Desplazamiento en diagonal del objeto tangible en Libre

Las series de los grados de la BB y del modelo del objeto tangible son prácticamente igual, así que para hacer más sencilla la gráfica. Estas series se encuentran en torno a las series de los grados esperados. Las del eje X son muy parecidas, mientras que la del eje Y se encuentra un poco por encima del esperado y el las del eje Z están un poco por encima. Aún variando un poco, se obtienen los valores dentro del rango esperado.

6.63: Gráfica de los valores que toman los ejes en el desplazamiento diagonal en Libre



8) Salir de Libre

Basado en el caso de uso 3.33. Desde la escena de Libre como se muestra en la pantalla *1* de la figura 6.64, el usuario puede volver al menú principal. Para ello, al poner el ratón sobre el botón *'Back'* como en la pantalla *2*, éste botón se hará grande. Al pulsarlo, hará una animación y pasará a la escena del menú principal en la pantalla *3*, haciendo la animación inversa que cuando se entró en la escena Libre, ocupando todo como en la pantalla *3* hasta llegar al menú principal en la pantalla *5*, pasando por la pantalla *4*.

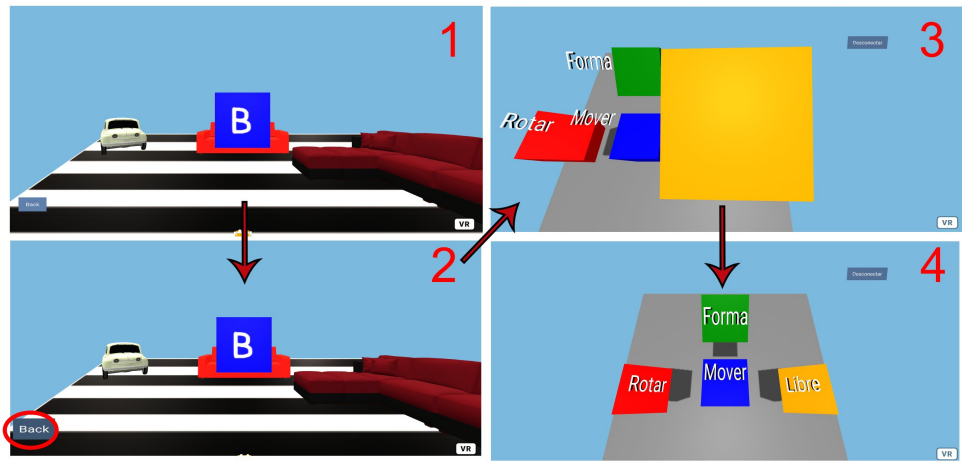


Figura 6.64: Salir de Libre

Capítulo 7

Conclusiones

Como se ha podido observar al con el análisis de los objetos tangibles y las plataformas de los objetos tangibles, no hay prácticamente ninguna orientada a solucionar los problemas del entendimiento como se plantean en este proyecto, aunque sí que hay algunos cercanos a ello orientados a la diversión del usuario.

En relación al objetivo principal, crear un objeto tangible que es imitado por su copia modelada en la pantalla para simular que el objeto físico está dentro del ordenador si se ha cumplido. Además de la posibilidad de cambiar de forma el objeto tangible para abarcar más objetos. La cuestión de si este proyecto es capaz de ayudar o no a mejorar las habilidades espaciales, de la percepción de la profundidad y el reconocimiento de objetos tridimensionales en pantalla, puede depender de cómo lo perciba cada persona.

Podría afirmarse que el reconocimiento de objetos si se ha conseguido gracias a poder ver todos los lados del objeto en la pantalla mientras se rota el objeto físico, pero afirmar la mejora de la profundidad y del entendimiento de entornos tridimensionales en pantalla todavía requiere más experimentación. Mi experiencia personal es que me ha ayudado, pero falta que más usuarios con problemas de percepción de la profundidad lo usen para afirmarlo con seguridad.

Si hay un entorno tridimensional dentro de la pantalla por el que moverse, alejando o

acercando el objeto que representa al objeto físico, pero la diferencia entre crear el entorno y conseguir que una persona pueda afirmar que ahora percibe mejor estos entornos o la profundidad, puede ser simplemente debido a que se ha acostumbrado a utilizarlo.

En cuanto al objetivo secundario, de ser capaz o no de crear un entorno ampliable de forma sencilla y que pueda servir para proyectos que hagan uso de objetos tangibles, parece cumplirse. Se han definido elementos genéricos de escenas y se han particularizado para el caso de estudio. No obstante, no ha dado tiempo a desarrollar aplicaciones concretas. Se deja para trabajo futuro.

El desarrollo de HITO, ha requerido el trabajo en diferentes entornos como son la BeagleBone, y el desarrollo en el ordenador para la interfaz y el backend, teniendo en cuenta las limitaciones físicas y el uso de un objeto físico.

7.1. Trabajo futuro

Teniendo el proyecto terminado, se echan en falta algunas funcionalidades y cuestiones para mejorarlo en un futuro. A continuación se exponen algunos de los puntos que podrían cambiarse y añadirse:

- Hacer el sistema del objeto tangible inalámbrico, es decir, quitar la necesidad de utilizar el cable de corriente para eliminar la limitación física añadiendo una batería. El hecho de tener el cable ha condicionado el proyecto para ciertas partes referentes al movimiento.
- Añadir la posibilidad de cambiar de escenario de objetos en el momento de estar en la escena con más objetos a parte del objeto tangible, como si fuese la elección de formas del objeto con las flechas de izquierda y derecha, para plantear nuevos retos sin tener que configurarlas a través del código.

- Crear nuevas carcasas para la BeagleBone, dado que solo está la del cubo alargado y si se cambia de forma, no hay cómo cambiarla en el objeto físico.
- Mejorar la calidad de las carcasas haciéndolas con una impresora 3D, porque ahora mismo la que hay es de cartón y puede mojarse o romperse por un golpe fuerte.
- Utilizar dos objetos tangibles de forma simultánea para poder mejorar la sensación de profundidad del sistema, además de poder simular que estos objetos son unas pinzas o manos, por ejemplo.
- Añadir nuevos sensores para poder hacer mediciones más precisas o para medir nuevas variables para nuevas funcionalidades.

Bibliografía

[1] AG, L. (2018), 'Tor - tangible object recognition [vídeo online]', Video TOR. [Consulta: 15-Nov-2019].

[2] Alex Kipman, M. (2020), 'marketplace.xbox', Kinect for Xbox 360. [Consulta: 15-Nov-2019].

[3] Amazon (2020), 'Aws (amazon web services)', Cloud9. [Consulta: 17-Feb-2020].

[4] Analog Devices, I. (2020), 'Analog devices, data sheet adxl345', ADXL345 Datasheet. [Consulta: 12-En-2020].

[5] Anderson, D., Frankel, J. L., Marks, J., Agarwala, A., Beardsley, P., Hodgins, J., Leigh, D., Ryall, K., Sullivan, E. and Yedidia, J. S. (2000), Tangible interaction + graphical interpretation: A new approach to 3d modeling, in 'Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques', SIGGRAPH '00, ACM Press/Addison-Wesley Publishing Co., USA, p. 393–402. [Consulta: 05-Jul-2019].
URL: *https://doi.org/10.1145/344779.344960*

[6] Apple (2019), 'Augmented reality', arkit. [Consulta: 20-Abr-2019].

[7] Arduino (2019), 'Arduino', arduino.cc. [Consulta: 20-Abr-2019].

[8] BBVAOPEN4P (2015), 'Mejores alternativas a arduino: del do it yourself al internet de las cosas', BBVAOPEN4P. [Consulta: 20-Abr-2019].

[9] Buxton, B. and Ubivid, U. U. (n.d.), 'Electronics, 3.27 (no. 632), 187-195. ubiquitous media and the active office', Digital Desktop. [Consulta: 05-Jul-2019].

[10] Cooper, J., Feldman, J. and Medlin, D. (1979), 'Comparing stereoscopic performance of children using the titmus, tno, and randot stereo tests', Journal of the American Optometric Association **50**(7), 821—825. [Consulta: 07-Feb-2020].
URL: *http://europepmc.org/abstract/MED/500993*

[11] Darnell, B. (2020), 'Tornado web server', tornado.org. [Consulta: 26-En-2020].

[12] deconceptos ((n.d)), 'deconceptos', tangible. [Consulta: 20-Ag-2019].

[13] donmccurdy (2020), 'aframe-physics-system', aframe-physics-system. [Consulta: 5-May-2020].

[14] El Saddik, A. (2018), 'Digital twins: The convergence of multimedia technologies', IEEE MultiMedia **25**(2), 87–92. [Consulta: 07-Dic-2019].

[15] Foundation, M. (2019a), 'minnowboard', minnowboard.org. [Consulta: 20-Abr-2019].

[16] Foundation, P. S. (2019b), 'Python', https://www.python.org/Python. [Consulta: 20-Jun-2019].

[17] Foundation, T. R. P. (2019c), 'raspberrypi', raspberrypi.org/. [Consulta: 20-Abr-2019].

[18] Google (2019), 'Arcore', https://developers.google.com/ar/. [Consulta: 20-Abr-2019].

[19] Gosling J, Sun Microsustems, O. (2019), 'Java software oracle', oracle java. [Consulta: 10-Abr-2019].

[20] Holowaychuk, T. (2020), 'Express', expressjs. [Consulta: 10-Dic-2019].

[21] Howard, Ian P.; Rogers, B. J. (1995), Binocular Vision and Stereopsis (Oxford Psychology Series), OUP USA. [Consulta: 07-En-2020].

[22] I., L. (2013), 'Cuatro alternativas a arduino: Beaglebone, raspberry pi, nanode y waspmote', Blogthinkbig.com . [Consulta: 20-Abr-2019].

[23] InvenSense, T. C. (2020), 'Tdk corporation invensense, data sheet itg3200', InvenSense itg3200. [Consulta: 13-En-2020].

[24] J, E. (2019), 'Etienne j', AR. [Consulta: 20-Abr-2019].

[25] Kridner, J. (n.d.), 'Beagleboeard.org', beagleboard. [Consulta: 20-Ag-2019].

[26] L., C. G. (2017), 'Dibujo técnico: tipos de perspectivas - mvblog', MVBlog . [Consulta: 10-Jun-2020].

[27] M., K. (2019), 'Openscad', OpenSCAD. [Consulta: 20-Abr-2019].

[28] McGookin, D., Robertson, E. and Brewster, S. (2010), Clutching at straws: Using tangible interaction to provide non-visual access to graphs, in 'Proceedings of the SIGCHI Conference on Human Factors in Computing Systems', CHI '10, Association for Computing Machinery, New York, NY, USA, p. 1715–1724. [Consulta: 25-Dic-2019].
URL: *https://doi.org/10.1145/1753326.1753583*

[29] MicTome (2020), 'tangible tfm', tangible TFM.

[30] Netscape Communications, F. M. (2020a), 'javascript.com by pluralsight.', javascript. [Consulta: 20-Dic-2019].

[31] Netscape Communications, F. M. (2020b), 'Mdn web doc mozilla', websocket. [Consulta: 26-En-2020].

[32] Nintendo (2020), 'Nintendo labo', Nintendo Labo. [Consulta: 10-Nov-2019].

[33] Node.js (2020), 'Node.js', Node.js. [Consulta: 20-Dic-2019].

[34] Osmo (n.d.), 'playosmo', Osmo. [Consulta: 07-Oct-2019].

[35] Playstation (2020), 'Playstation', Playstation Move. [Consulta: 15-Nov-2019].

[36] Pruitt, J. and Adlin, T. (2010), The Persona Lifecycle: Keeping People in Mind Throughout Product Design, Interactive Technologies, Elsevier Science. [Consulta: 15-En-2020].
URL: *https://books.google.es/books?id=Ct7kU5kO_T8C*

[37] (psiphi75), S. W. (2016), 'github gyroscope itg3200', https://github.com/psiphi75/gyroscope-itg3200. [Consulta: 10-En-2020].

[38] QuiverVision (2016), 'Quiver vision', Quiver. [Consulta: 07-Oct-2019].

[39] Roosendaal, T. (2020), 'blender.org', blender.org. [Consulta: 10-Feb-2020].

[40] Rosendo, I. G. (2017), '¿estás seguro de que ves bien en 3 dimensiones? te explicamos cómo saberlo', BBC Mundo Salud . [Consulta: 10-Abr-2020].

[41] seeed (n.d.a), 'wiki.seeedstudio', Grove System. [Consulta: 20-Dic-2019].

[42] seeed (n.d.b), 'wiki.seeedstudio', BeagleBone Green Wireless Wiki. [Consulta: 20-Oct-2019].

[43] SeeedStudio (n.d.a), 'seeedstudio.com', SeeedStudio ToF Sensor. [Consulta: 07-Feb-2020].

[44] SeeedStudio (n.d.b), 'seeedstudio.com', SeeedStudio. [Consulta: 07-Feb-2020].

[45] Sensics (2014-2016), 'Osvr', osvr.github.io/. [Consulta: 20-Abr-2019].

[46] S.L, L. C. D. (2019), 'Libelium', libelium.com. [Consulta: 20-Abr-2019].

[47] Solutions, E. (2020), 'Tor – tangible object recognition', TOR. [Consulta: 10-Nov-2019].

[48] Stroustrup B, N. B. L. (2019), 'scpp.org y open-std', isocpp.org/ open-std.org. [Consulta: 20-Abr-2019].

[49] Supermedium, Google, W. (n.d.), 'Aframe', aframe.io/ A-Frame.

[50] Tangible Virtuality: Towards Implementation of the Core Functionality (2008), Vol. Volume 3: 28th Computers and Information in Engineering Conference, Parts A and B of International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. [Consulta: 05-Jul-2019].
URL: *https://doi.org/10.1115/DETC2008-49174*

[51] Technologies, U. (2019), 'unity', unity.com. [Consulta: 20-Abr-2019].

[52] Uffenbeck, J. E. (1999), Microcomputers and Microprocessors: The 8080, 8085, and Z-80 Programming, Interfacing, and Troubleshooting, Prentice-Hall, 1985. [Consulta: 07-En-2020].

[53] Vishwanath, D. (2013), Immersion, tangibility, and realism: explaining the qualitative experience of stereopsis, in A. J. Woods, N. S. Holliman and G. E. Favalora, eds, 'Stereoscopic Displays and Applications XXIV', Vol. 8648, International Society for Optics and Photonics, SPIE, pp. 188 – 209. [Consulta: 10-Nov-2020].
URL: *https://doi.org/10.1117/12.2004902*

[54] You, B.-J. and Ahn, S. (2020), 'Tangible tele-meeting system with a visually augmented robot'. [Consulta: 5-Jul-2019].

Apéndice A

Introduction to tangible objects

A.1. Introduction

This project is motivated by the difficulties that some persons can have to understand how objects with specific shapes can exist in an environment that in advance is plain, like it's a computer's screen, and how a three-dimensional space can exist in them. This type of problems can be in both children and adults that start intreacting with computers, juyst like persons with speech therapy problems or persons with sight problems. This sight problems are named as sight stereoscopic problems, binocular disparity or a deficiency in depth preception, what it means lack of stereopsis[21] [53], that is to say, a deficiency development of non-innate capacity to create three-dimensional images with their eyes, what it means lack of capacity to correctly preceive depth and the perception of three dicemnsions in unfavourable conditions. This can imply difficulties when the persons try to drive with security, make jobs or exercises that require good precision or enjoy and work in videogames industry.

In day to day, without having this capacity, the brain develop some methods by using ilumination, shadows, reference objects, volume and size to suply this problem. A clear example of this problems are when a person has to tilt it's head to focus and to correctly preceive some objects. However, when it comes to applying this methods in 3D environments, it doesn't work in the right way because this environments don't must have to have shadows, ilumination, the distances and sizes could not be the used ones in real world and the depth

can be misleading.

To solve this problem is proposed to take the advantage of Tangible Object concept [12]. This type of elements could provide more intiotive way to translate the things that happen on the real world to a 3D scene and vice versa.

A.2. Objectives

The realization of this project have as main objective the creation and utilization of a tangible object that, as it name says, is an object that can be physicaly touched and manipulated and allows to interactuate in one way or another with a digital element. Like this, with a tangible object a three-dimensional one can be represented in a computer's screen with the purpose of improve or develop a better understanding of three-dimensional evironments. It also can help to improve spatial abilities and the object recognition in this environments.

With this tangible objects it tries to simulate the reality in a computer and check the depth capacity and geometry and try to improve this capacities by doing a realtion between object and model in computer's screen.

With this relation between tangible object and screen's model, the main objective is to search help to handle depth, distances and recognition capacity of real shapes in three-dimensional environments by the similarity between this tangible objects and models and the model's imitation of shape and moves. With the possibility of tangible objects to change it's shape, the inmersion can be increased and allow a better way depth understanding.

To achieve this objective, the tangible object should be allowd to give it's movement information and send them to the computer at it's connected without block it,

As a secondary objective, it searchs to create a system that allow a more simple development of solutions that make use of a physical object and a virtual model. For which the

system needs a modifiable and reusable framework.

As resume, the objectives are:

- Create a low cost tangible object.
- Make that the tangible object and it's screen representacion have the same shape and their shape can change.
- Gesture and movement tangible object identification to make it's screen representation imitate those gestures and movements and create an inmersion relation.
- Create a framework to speed up the development of applications that make use of tangible objects.

With objectives defined, the project try to relation physical objects with it's model on computer's screen to help to improve. With that idea, the system is called **HITO** (*Help to Improve with Tangible Objects*).

A.3. Workplan

This Document details the development of a system with tangible object, a physical object is reflected in a three-dimensional enviromnent within a computer screen, allowing to reflect the movements of the tangible object in it's digital reflect.

El desarrollo del proyecto se ha definido en dos partes, primero una definición del producto y luego una explicación del mismo definiéndolo en sus puntos más básicos. Para ello, se han definido las siguientes fases:

The development of the project has been defined in two parts, first one is the producto's definition and later an explanation of it by defining it in it's basics points. To achieve this, it has been defined the following phases:

1. **State of art:** On this phase it is going to be known other works related to tangible objects. It will cover and analysis of compentence split in projects that use objects to interact and in platforms that use them. Also it will be analyze the unchoosen technologies.

2. **System requirements:** This phase contains all the user research and the project desing. The users are created to reflect the objective public of this product.

    2.1. **User research:** It searchs the posible users parting of the initial premise of the project, persons with three-dimensional understanding problems.

    2.2. **Modeling:** Parting of the last data, fictional persons are created to represent the objective users and there's purposed sitiations in which the system could be usefull, getting the system objectives.

    2.3. **Requeriments:** Parting of the objective users and objectives found, the system requeriments are defined.

    2.4. **Product description:** With all the last knowledge, HITO's general concept is created. The interaction framework that allows to create the visual and physical prototype are defined. It's done with incremental iterations to increase prototype fidelity.

    2.5. **Use cases:** With the framework definded, the system posibilites to interact with the tangible object are defined.

    2.6. **Technologies:** The system technologies are chosen, including sensors.

When the prodyct is defined, it's going to proceed to build and test it:

1. **System architecture:** With all the last data, it proceeds to systen development, describing the architecture and it's components.

2. **System Implementation:** After the architecture, the used implementation is going to be explainted.

3. **Experimentation:** Detailed descriptions are proposed to describe how it should be used the system in all moments.

A.4. Document structure

To structure the workplan described in the last section, a document structure that gather all the important points like chapters and sections of them is created. In adition, there will be added appendixes with relevant information for the project like aditional documents that aren't within the workplan chapters development.

This chapter is about to give the necessary information to understand the reasons and objectives to develop the project.

In the chapter 2 is studied, how it's said in the section 1.3, the technologic advance related to the project. Splitting this in proyects with tangible objects in the section 2.1 and in platforms that use them in the section 2.2. Moreover, the discarded technologies when the project was created are shown in section 2.4.

In the chapter 3 the system requirements are defined by an user research in section 3.1 with a modelling in section 3.2, the fictional persons, the requeriment establishment in section 3.3 and the product definition in section 3.4. Furthermore of system use cases in section 3.4.5. Also the chosen technologies are established in section 2.5.

In chapter 4, parting of state of art and systen requeriment results, the system development starts, where its explained the architecture in section 4.1 and the desing of it in section 4.2.

In a chapter apart, there's the chapter 5, the implementation where the full system components are explained, splitting computer development in stection 5.1, from the tangible object in section 5.2.

As last chapter, the 6 of workplan, the experimentation with examples of using the system developed are explained. In the chapter 7, the last one of the statement, obtained conclusions at the end of the project development are detailed and future work to improve or tu add new elements to the project are thinked.

In the appendix there are the manuals to the system display, in appendix C, to know how to display it, and the user one to know how to use it for both user objectives and secondary ones in appendix D.

Apéndice B

Conclusions

As it has been observed with the the analysis of tangible objects and the platforms of tangible objects, there are practically none that si oriented on solve the problems of understanding like it's proposed in this project, although ther are some near to this with the focus on make it enjoyable for the user.

Related to the main objective, making a tangible object which is imiatated by it's modeled copy in the screen to simulate that it's the physical object into the computer, it's accomplished. In adition of the posibility to change the shape of the tangible object to be more objects. The question of whether this project is able to help or not in the improve of spacial abilities, the depth pereception and the three-dimensional objects understanding on screen, can depend on how the person preceive it.

It could be stated that the understanding objects on screen it's accomplished thanks to be allowed to see all the object faces on screen while the physical object is rotated, but assure the improve of depth and of the understanding in three-dimensional environments on screen still require more experimentation. My personal experience is that it helped me, but it's necessary that more users with depth preception problems it to confirm it with security.

There is a three-dimensional enviromnment in the screen where you can move, zooming in or zooming out the physical object's representation, but the difference between make

the environment and achieve that a person can confirm that she can preceive better this environment or depth, it can be simply cause she is used to use it.

As for the secondary objective of been able or not to create an expandable environment in a simple way and that it can be usefull for other projects which make use of tangible oblects, it seems like it's accomplished. There are defined generic scene elements and a case study is particularized. However, has no gibben time to develop comprete applications. It's left for future work.

HITO's development has required the work if differents environments like the BeagleBone, and the development in the computer fo the interface, taking in account the physical limitations and the use of a physical object.

B.1. Future work

Having the project finished, there are some missing functionalities and questions to improve it in the future. Bellow thare are some points that could be changed or added:

- Make the tangible object system wireless, what means in removing the need to use the power cable to delete the physical limitation by adding a battery. The fact of having the cable has conditioned the project in some ways related to the movement.
- Add the posibility of change the objects scenario in the time where you are in the scene with more objects in adition to the tangible object, like it's the shape choice of the object with the left and right arrow to propose new challenges without having to configurate them through code.
- Create new BeagleBone's shells cause at the moment theres only the long cube and if the shape is changed, there is no way to change it physically.
- Improve the quality of shells by making them with a 3D printer because right now, the exisitng one its made of carton and it can be ruined by betting wet or be hard hit.

- Make the use of two tangible objects at the same time to improve the feeling of depth, in adition to be allowed to simulate that this objects are hands or tongs, for example.

  Utilizar dos objetos tangibles de forma simultánea para poder mejorar la sensación de profundidad del sistema, además de poder simular que estos objetos son unas pinzas o manos.

- Add new sensors to allow more precise measurement or measuring new variable for new features.

Apéndice C

Manual de despliegue

En este apartado se explicará qué se necesita para para instalar el sistema: Tangible y los pasos necesarios para conseguir su correcto funcionamiento.

C.1. Requisitos previos

Para poder ejecutar bien el proyecto, es necesario tener los siguientes elementos:

- **Ordenador:** el ordenador tiene que tener como sistema operativo Ubuntu 20.04 y al menos 4GB de memoria RAM.
- **BeagleBone Green Wireless**[1]**:** esta placa tiene que ir acompañada de los elementos siguientes:

  Tarjeta de memoria 8GB[2]

  Acelerómetro ADLX345

  Giroscopio ITG3200

  Adaptador Hub I2C: necesario para poder utilizar ambos sensores.

[1] Solo necesaria para la ejecución del objeto tangible
[2] Solo necesaria si se elige no sobre escribir el SO original de la BeagleBone

C.2. Preparación del entorno

Para poder utilizar el proyecto, es necesario preparar tanto la BeagleBone como el ordenador. Lo fundamental es descargar de github los archivos del proyecto. Se puede descargar directamente desde este enlace y descomprimirlo en la carpeta que se desee o utilizar el comando *git* con *'git@github.com:MicTome/tangible—TFM.git'*. Una vez hecho esto, se preparan las dos partes del proyecto.

Preparación del ordenador

Para poder hacer funcionar esta parte del proyecto, es necesario descargar e instalar *nodejs*, intérprete de JavaScript para poder usar bien el servidor de la aplicación con la interfaz. El proyecto utiliza la versión 'v10.19.0'.

Para instalar nodejs, hay dos formas de hacerlo, la primera descargando e instalándolo desde su página web (imagen C.1) y descargar el archivo recomendado para la mayoría. En el momento de la realización de este manual, la versión es la 'v12.17.0 LTS'.

Figura C.1: Página de node

La segunda y la que se ha utilizado para el proyecto es abrir una terminal y con permisos de administrador, ejecutar los siguientes comandos:

1. **'apt update'** para actualizar los repositorios.

2. **'apt install nodejs'**

3. **'apt install npm'** el administrador de paquetes de nodejs.

Cada comando puede tardar un tiempo en ejecutarse. Cuando se instale npm, hay que hacer uso del nuevo comando para instalar 'express', porque el proyecto lo usa junto con nodejs para arrancarlo. El comando a utilizar es: **'npm install express'**.

Puede dejar de utilizarse el usuario con permisos de administrador dado que ya no se va a instalar nada más. Después de estas instalaciones, hay que extraer el archivo descargado desde github.

Finalizados los pasos anteriores, ya se tiene la primera parte del proyecto instalado correctamente.

Preparación de la BeagleBone

Para conseguir hacer funcionar la placa para que funcione el proyecto, es necesario actualizar el sistema de la misma. Para ello, se va a utilizar un sistema Debian, en caso de que la placa que se esté usando tenga el mismo y no esté actualizado, lo más probable es que se deba descargar la imagen para quemarla en la placa.

En el caso de este proyecto, se utiliza la imagen de **'Debian 9.9 IoT'** del 03-08-2019. Para descargar esa u otra se accede la página de beagleboeard.org, figura C.2, donde actualmente tiene otra versión y más abajo se pueden descargar imágenes antiguas.

Figura C.2: Imagenes de BeableBone

Existen dos maneras de instalar el sistema operativo recientemente descargado, la primera es instalarla directamente en la memoria de la BeagleBone sobreescribiendo la que ya tenía, como se explica aquí, o la segunda forma que es la que se ha utilizado, quemando la imagen del SO en una tarjeta de memoria externa. Para hacerlo, hay que descargar el programa específico para imágenes en memorias extraíbles, balenaEtcher.

Una vez descargado y descomprimido, se obtendrá una aplicación portátil, es decir, que no es necesario instalarla. En este momento será necesario introducir la tarjeta de memoria con su adaptador en su ranura en el ordenador. Al abrir el programa, se mostrará la interfaz como en C.3 donde se tiene que buscar o arrastrar la imagen descargada, seleccionar la unidad donde instalarla y darle al botón 'Flash'.

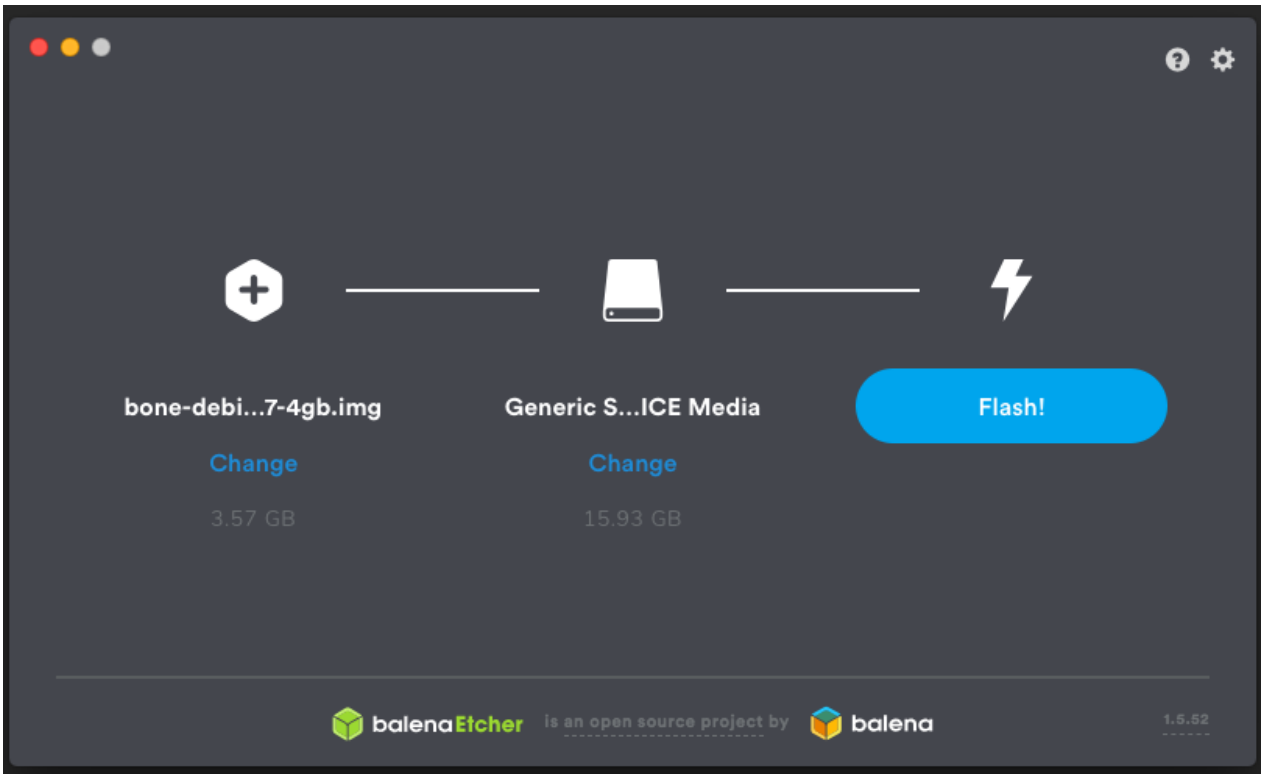


Figura C.3: Balena

Una vez hecho esto, es hora de configurar el nuevo sistema de la placa. Para ello, antes de enchufarla, es necesario conectarle los sensores a través del Adaptador I2C y conectar éste en el 1 de la figura C.4, que es el puerto I2C, meterle la tarjeta en su ranura como se ve en el 2 y, al conectarla, mantener apretado el botón del 3 de la figura hasta que los leds del 4 comiencen a parpadear. El ordenador tardará un poco en detectar la BeagleBone.

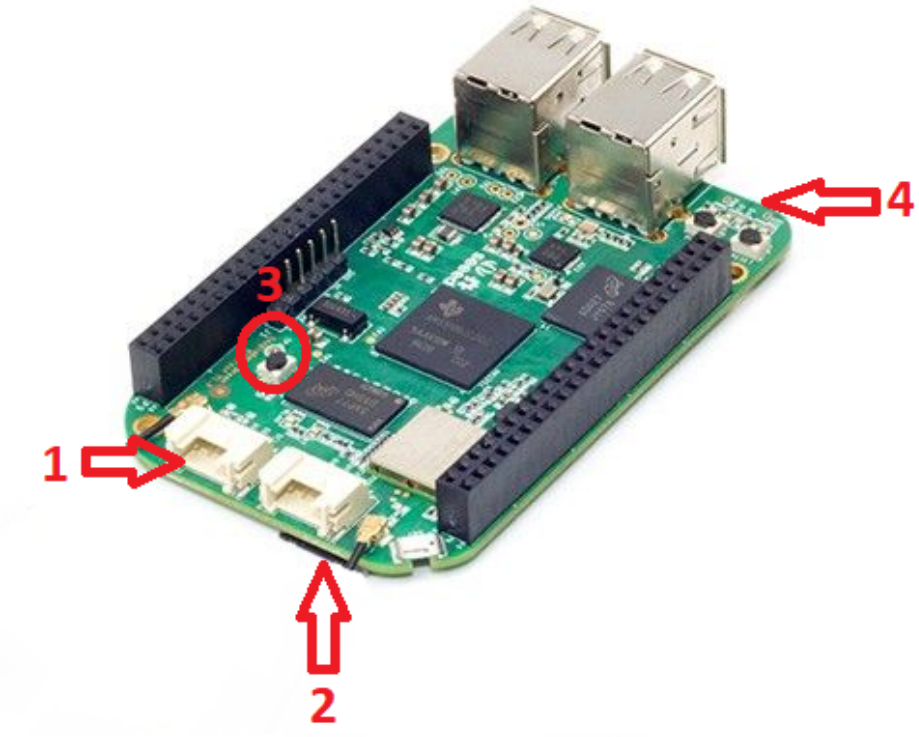


Figura C.4: Beagle

Si se tiene una caja para simular un objeto tangible, es recomendable que esté la placa primero en ella y luego proceder a las conexiones.

Una vez conectada, hay que acceder a la dirección de la beagle y se accede buscando la dirección IP de la misma en el navegador. Su dirección es la **192.168.7.2** y si todo está correcto, se abrirá cloud9, como en la figura C.5, un entorno de desarrollo.

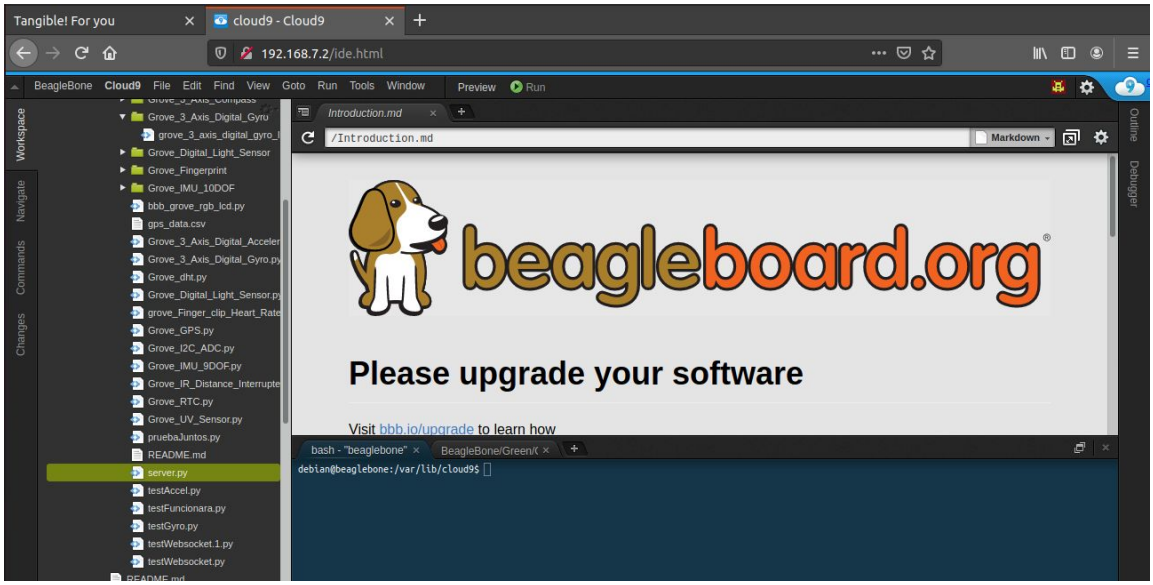


Figura C.5: Cloud9

En la parte inferior se puede ver una consola del sistema, la cuál hay que utilizar para los siguientes pasos. Lo primero, es necesario tener permisos de administrador, por lo que introducimos el comando **'sudo su'** e introducimos la contraseña por defecto que es **'temppwd'**, sin las comillas.

Siendo administrador, hay que conectarse a internet para actualizar e instalar los paquetes necesarios para el proyecto. Como en la figura C.6 se utiliza el comando **'connmanctl technogies'** para comprobar si el wifi está conectado. La línea 'Connected = True' indica que sí lo está.

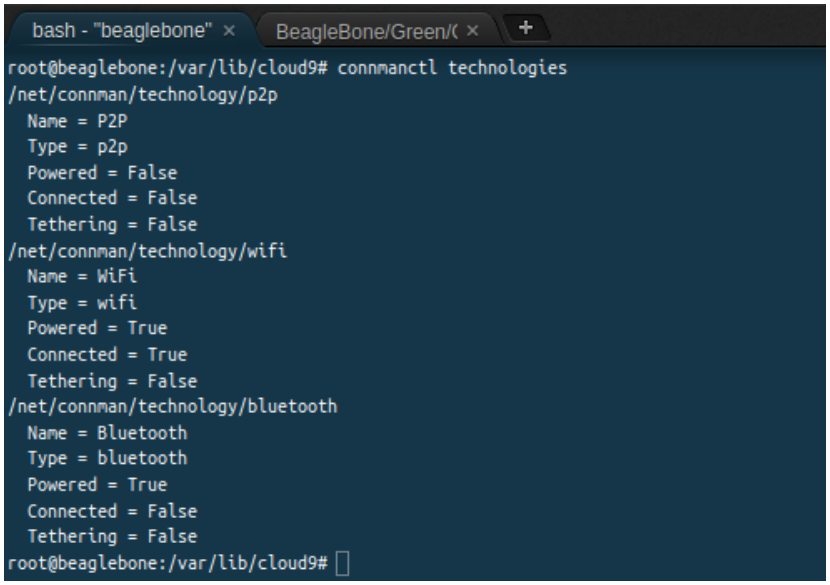


Figura C.6: connmanctl technologies

Si no está conectado, hay que usar el comando **'connmanctl enable wifi'** y comprobar que se activó con el anterior otro comando.

Ahora es necesario utilizar el modo interactivo de ConnMan usando **'connmanctl'**. En este modo hay que escribir **'scan wifi'** que escanea las redes wifi que estén al alcance pero no las muestra. Para mostrarlas hay que usar **'services'** como en la figura C.7, Nos importa la parte derecha de las redes que se muestran.

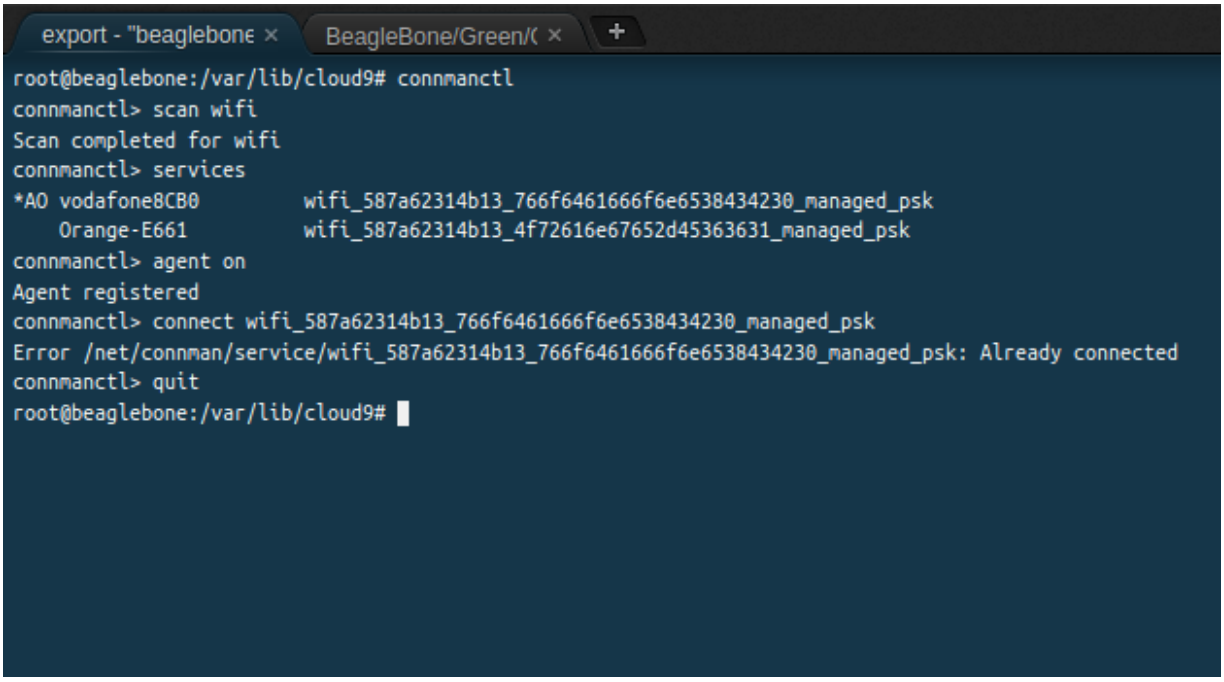


Figura C.7: Conectar wifi

Luego hay que usar **'agent on'** para poder recibir peticiones del usuario y utilizar el comando **'connect wifi_right-name'** para hacer la conexión con ese wifi. Utilizando **'quit'** se sale del modo interactivo de connmanctl.

Con esto hecho, hay que actualizar la beagle con **'apt-get update'** y **'apt-upgrade'**. Una vez actualizada, hay que instalar los paquetes necesarios con los siguientes comandos:

1. **'apt-get install python3'**
2. **'apt-get install python-pip'**
3. **'apt-get install python-setuptools'**
4. **'apt-get install python-smbus'**
5. **'pip install Adafruit_BBIO':** librería con herramientas para utilizar sensores.
6. **'pip install adafruit-circuitpython-adxl34x':** librería concreta para utilizar sensores adlx34x como el acelerómetro utilizado en el proyecto.

7. **'pip install tornado':** websocket y generador de bucles.

Una vez hecho esto, solo es necesario arrastrar el archivo de dentro de la carpeta 'beagle' del proyecto descargado a la parte izquierda de cloud9, donde se muestran archivos de ejemplo. Si todo ha salido bien, ya se tiene la segunda parte del proyecto preparada para su funcionamiento.

C.3. Ejecutar sistema

Para ejecutar el sistema completo, hay que tener arrancadas ambas partes, tanto la del ordenador como la beagle. El proyecto está preparado para arrancar cualquiera de las dos antes, pero para que resulte más práctico, se empezará por arrancar la beagle.

En la beagle, en cloud9, hay que abrir el archivo y pasará a modo edición. En la parte de arriba, hay que pulsar el botón 'Run' con el icono verde. Se deberá abrir una nueva consola con el servidor arrancado.

Por la parte del ordenador, hay que ir a la carpeta 'tangible' del proyecto descargado y ejecutar el comando **'npm run serve'** que ejecuta node internamente para crear un servidor nodejs al que se conecta el proyecto.

Una vez hecho esto, abrir un navegador, recomendable Firefox o Google Chrome y buscar en la barra de direcciones **'localhost:3000'**, en ese momento se podrá interactuar con la interfaz del proyecto.

Figura C.8: Tangible

C.4. Uso del sistema

A partir de este momento, se usará tanto el ratón como la placa. Al clickar en el botón Start que aparece se mostrará el menú y desde aquí clickar en Rotar, Mover o Libre para interactuar con la placa, el objeto tangible.

En el menú se muestra arriba a la derecha un botón para conectar o desconectar de la placa, que sería el utilizado si se arrancase primero la parte del ordenador para conectar con la placa.

Apéndice D

Manual de usuario

En este capítulo se explicará como usar el sistema una vez se ha puesto en marcha, como se explica en el capítulo C Manual de usuario. Se va a explicar tanto para los usuarios objetivos como para los usuarios que quieran modificar el proyecto y añadirle más funcionalidades.

D.1. Usuario objetivo

En este apartado se va a explicar para aquéllos usuarios que harán un uso normal de la aplicación. Para poder interactuar será necesario el ratón del ordenador y el objeto tangible, la placa, que ya está conectada.

Teniendo ya abierto el navegador con la interfaz, lo primer que hay que hacer es darle al botón Start con el ratón y se pasará de la imagen izquierda a la de la derecha, que es el menú principal.

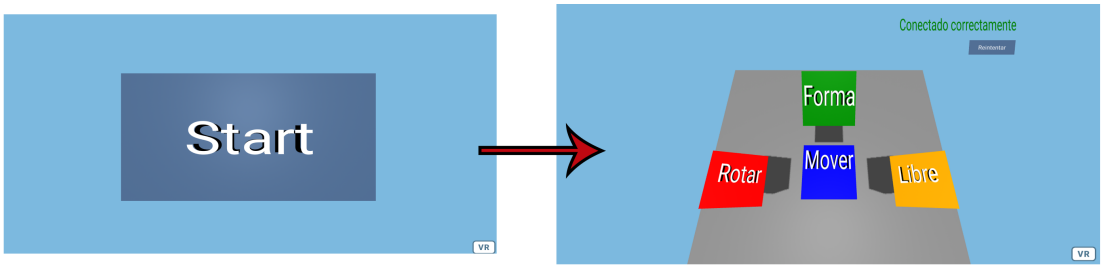


Figura D.1: Inicio a menú

Es posible que no se haya podido conectar al servidor, por lo que habrá que clickar en el botón de arriba a la derecha donde pone 'Reintentar' y un texto en rojo 'El servidor no responde'. En este estado, si se intentase acceder a Rotar, Mover o Libre, no se podría acceder porque las escenas de esos cubos se bloquean mientras el servidor no esté conectado, pero la de Forma si está disponible. Una vez clickado en 'Reintentar', debería pasar a poner un botón 'Reintentar' y un texto en verde que desaparece en un tiempo. Una vez conectado, habrá que esperar un par de segundos para que los sensores se calibren y se pueda usar correctamente.

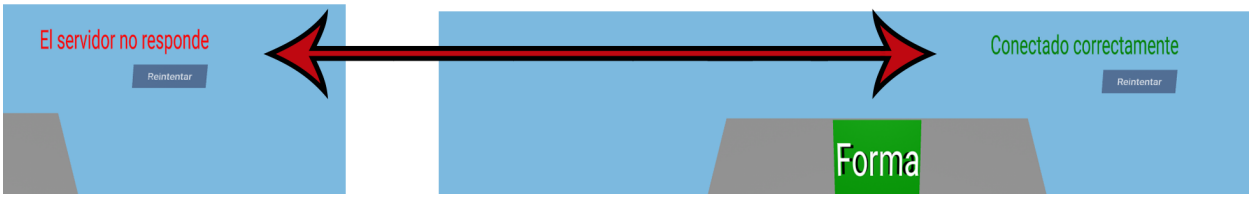


Figura D.2: Conexión con el servidor

En caso de que aún pulsando 'Reintentar' no se conecte con el objeto tangible, habrá que comprobar si está conectado, que debería de estar en otra pestaña. En esa pestaña, la consola de abajo debería mostrar un botón rojo de Stop. Puede ser que no se haya conectado

correctamente, por lo que pulsando ese botón una vez, se pondrá en verde con Start y habrá que pulsarlo nuevamente como en la imagen inferior. Nuevamente en la interfaz, habrá que darle a 'Reintentar' y se debería conectar correctamente.

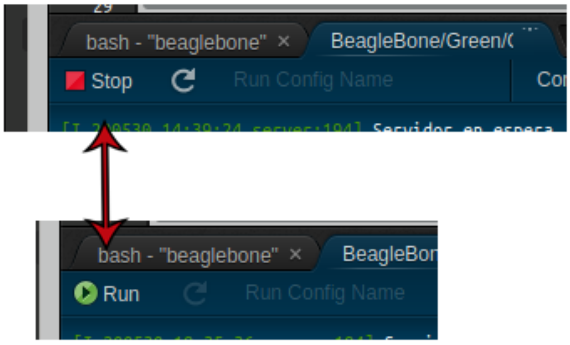


Figura D.3: Apagar/encender servidor

A partir de este momento, hay cuatro cubos en los que poder clickar. Lo que hará cada uno es lo siguiente:

- **Forma:** el cubo verde, al clickar en él se podrá ver cada una de las formas que puede adoptar el objeto tangible que representa al objeto físico. Aquí se pueden hacer varias cosas. Si se pulsa en el botón de abajo a la derecha donde pone 'Actualizar' los objetos del centro se actualizarán a los objetos que haya en la carpeta **'assets/models/'** del proyecto. Se pueden añadir más pero han de ser modelos .glb o .gltf con su .bin. Se pueden modelar desde blender.

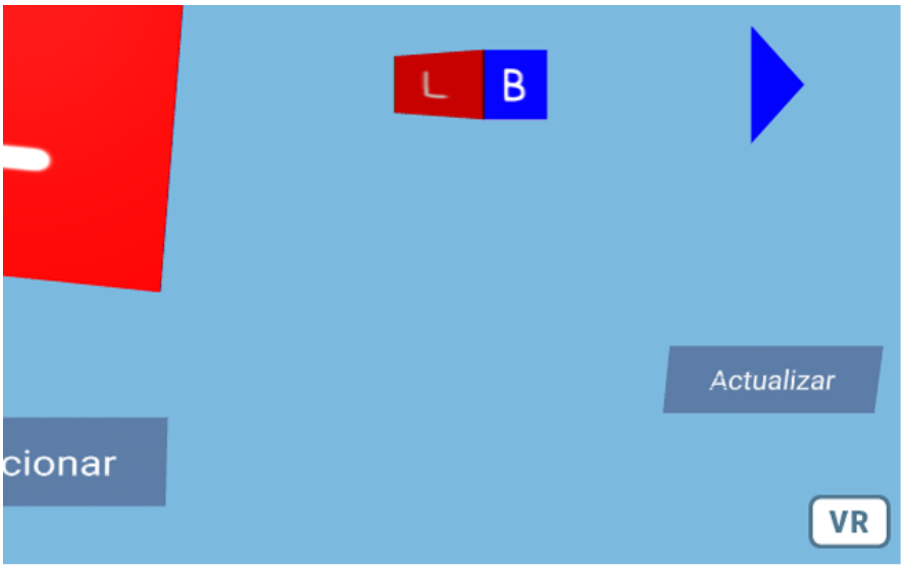


Figura D.4: Actualizar

Si se pulsan las flechas de los laterales, rotarán los objetos centrales para que en el centro quede la forma que se desea para el objeto tangible.

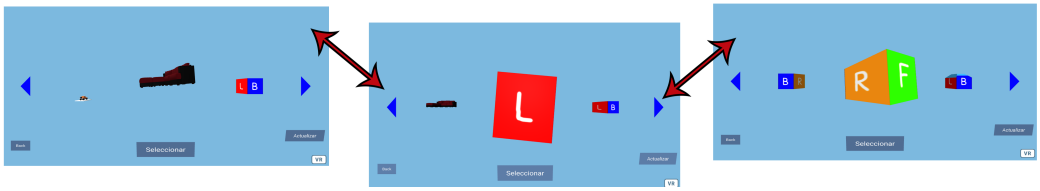


Figura D.5: Rotar el carrusel

Si se si se mantiene pulsado el objeto central, dejará de rotar por su cuenta y se rotará a gusto del usuario, El botón 'Back' y el 'Seleccionar' mueven la escena a la del menú, pero el segundo lo que hace es cambiar la forma de los objetos tangibles y de los cubos del menú para mostrar el cambio. Atención, una vez cambiada la forma de los cubos del menú, no podrán volver a su forma y color original.

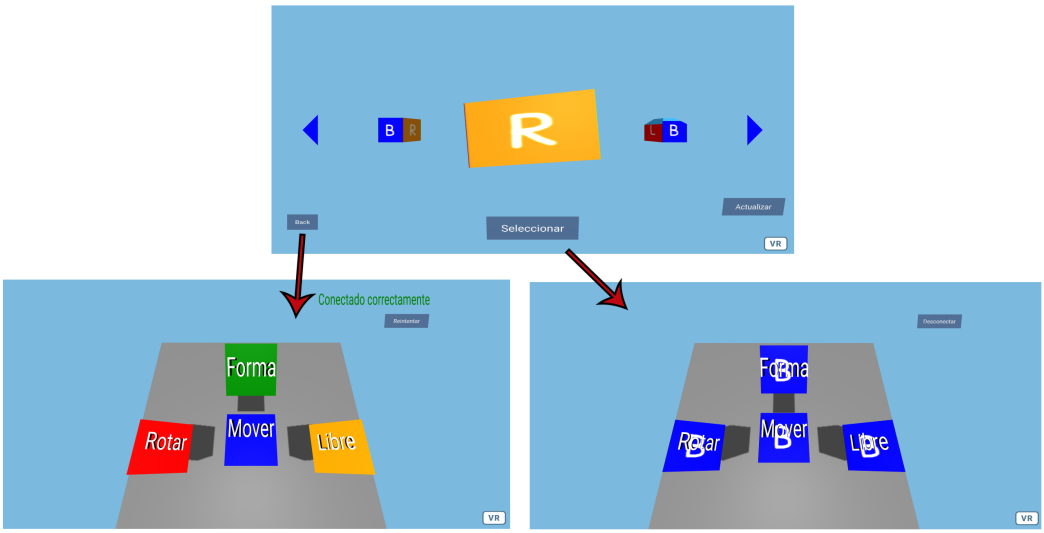


Figura D.6: Seleccionar o no objeto

- **Rotar:** aquí se mostrará sólo el objeto tangible y es donde se podrá rotar el objeto físico para que en ese objeto de la pantalla, se produzcan las mismas rotaciones. También se muestra un botón 'Back' para volver al menú principal.

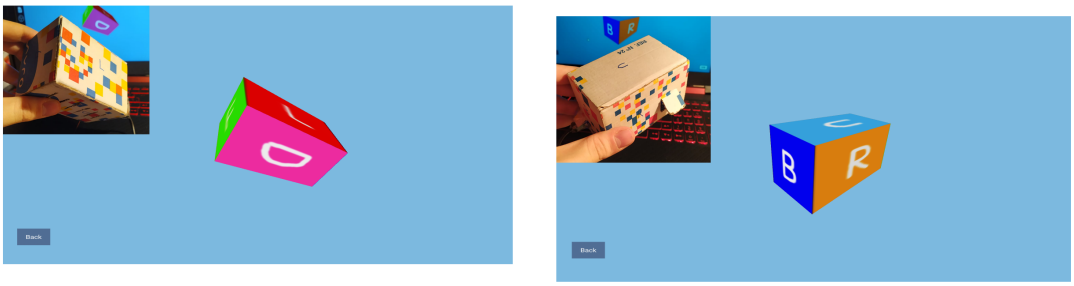


Figura D.7: Rotar

- **Mover:** en esta escena se muestra el objeto tangible y un conjunto de objetos interactuables. El objeto tangible no rotará con tanta precisión, solo imitará en ciertos grados el movimiento del objeto físico, pero en adición, se desplazará por la pantalla. En seis direcciones posibles.

  Inclinando el objeto físico a izquierda y derecha el objeto modelado se moverá a iz-

quierda y derecha respectivamente.

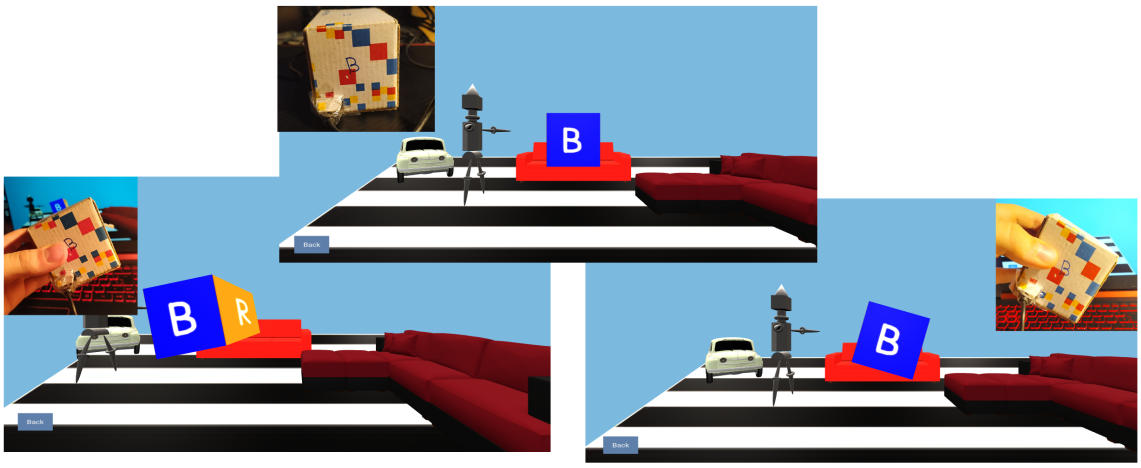


Figura D.8: Inclinación lateral

Inclinando el objeto hacia adelante, alejándose de la cámara, o hacia atrás, acercándose, se moverá en esa dirección.

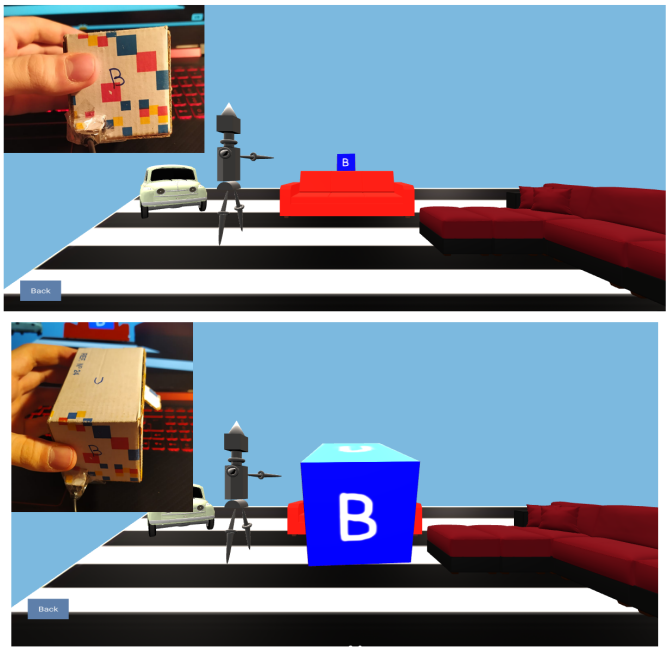


Figura D.9: Inclinación frontal y posterior

Si el objeto se inclina mucho hacia adelante, dejará de alejarse para comenzar a des-

cender. Si se inclina mucho hacia atrás, dejará de acercarse y comenzara a ascender.

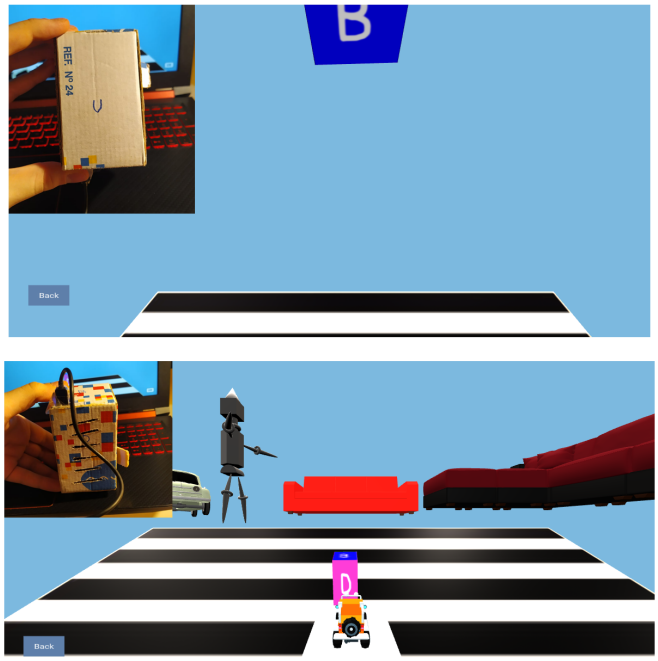


Figura D.10: Ascender y descender

Otra posibilidad que hay es la de tocar otros objetos de la escena, cuando esto ocurra, mostrarán un texto hasta que se deje de tocarlos. Algunos elementos realizarán algo a parte de mostrar un texto.

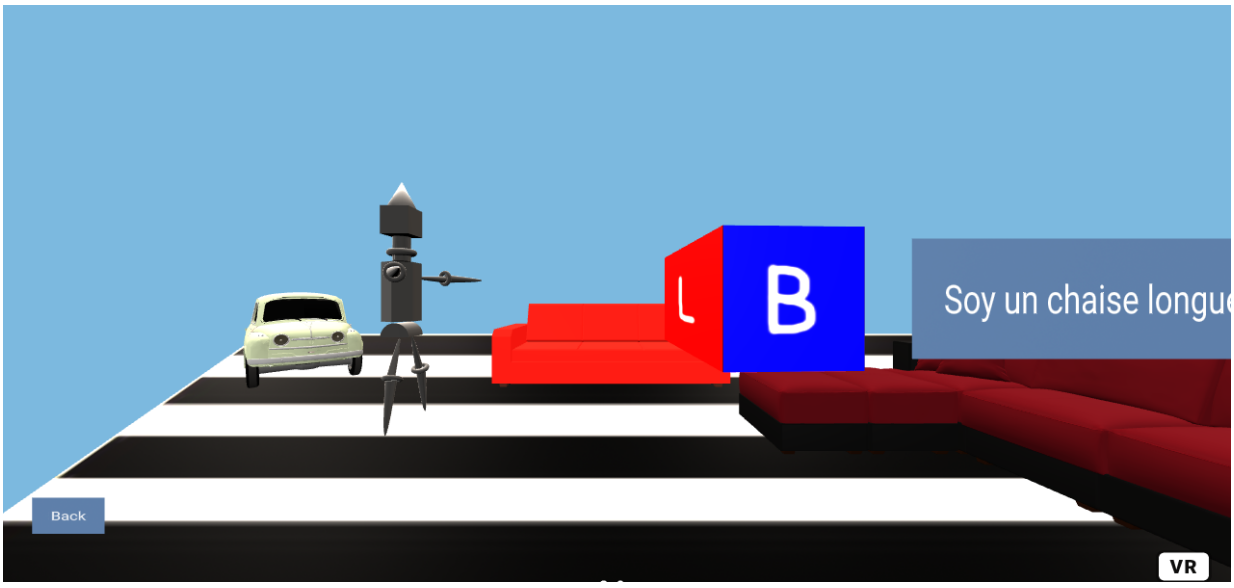


Figura D.11: Interacción

El botón de 'Back' siempre presente llevará al menú principal.

- **Libre:** en esta escena, se muestra el objeto tangible y más objetos. El objeto realizará las rotaciones del objeto físico y además se desplazará con las inclinaciones. La cámara sigue al objeto para no perderlo.

  El botón de 'Back' siempre presente llevará al menú principal.

D.2. Otros usuarios

Estos usuarios son los que se representan como aquéllos que no hacen uso del sistema para mejorar en sus problemas, sino que buscan añadir más funcionalidades al mismo para que sirva para otros fines.

Aquí se explicará cómo añadir algún elemento nuevo sobre el proyecto, ya sea un nuevo sensor o alguna escena nueva en la interfaz.

D.2.1. Nuevo sensor

Para añadir un nuevo sensor, hay que estar en la pestaña con cloud9. Si es necesario descargar un paquete nuevo, después de conectarse a internet como se explican en C.2, hay que instalarlo con la consola inferior. Recomendable con usuario administrador.

Abriendo el archivo server.py, hay que añadir el código necesario del sensor para poder leerlo. Para enviar los datos leídos, en la imagen D.12 se muestra una línea con el envío en formato JSON, donde se tendrían que añadir los nuevos datos.

Figura D.12: Envío en JSON

Luego habría que adaptar los métodos 'getSensorData()' y 'clearSensorVariables()' en 'methods.js' para que se lea la información nueva.

D.2.2. Nueva escena

Para añadir una nueva escena, primero habría que añadir un nuevo objeto en la parte de a-frame donde se encuentra el menú, como en la imagen, o asociarlo al mismo creando los objetos con a-frame en el script dentro de 'index.html'. En ambos casos habrá que añadir los listeners del objeto como en el script de 'index.html'.

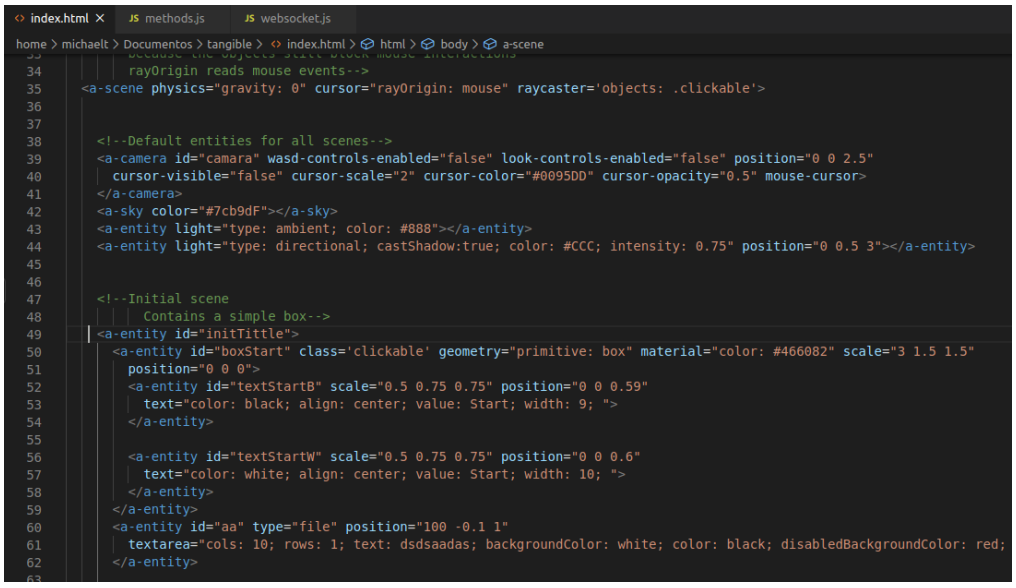


Figura D.13: a-frame

Para añadirle su funcionalidad habrá que crear su método en 'controller.js', en 'configSceneMethods.js' añadir el método para configurar la escena y en 'methods.js', la funcionalidad de la misma.

Si se quiere añadir una nueva funcionalidad, habrá que añadirla en 'methods.js'.

Apéndice E

Tablas del retraso entre los datos en Ubuntu y BeagleBone

En este anexo se encuentran los documentos de todas las tablas hechas en el capítulo 6 de la experimentación en las que ha sido necesario el uso sincronizado de los datos de la BeagleBone con los recibidos en Ubuntu.

Todas las tablas tienen 30 muestras y el mismo esquema: las dos primeras columnas son el tiempo en milisegundos en el momento de la ejecución, la siguiente es la diferencia en milisegundos de las dos primeras, la cuarta es el intervalo entre una toma de datos y otra, que es de **83** milisegundos para que no se solapen las tomas de datos, la penúltima es la diferencia entre la tercera columna y está última para conocer el retraso en cada muestra y la última es la media total de los retrasos.

E.1. Rotar

Tabla de retraso de rotar horizontalmente en Rotar

Tiempo en Ubuntu	Tiempo en Beagle	Diferencia	Intervalo	Diferencia	Media
1592586842833	1592586842742	91	83	8	24,06
1592586842942	1592586842825	117	83	34	
1592586843011	1592586842909	102	83	19	
1592586843094	1592586842991	103	83	20	
1592586843178	1592586843074	104	83	21	
1592586843261	1592586843158	103	83	20	
1592586843345	1592586843241	104	83	21	
1592586843429	1592586843323	106	83	23	
1592586843523	1592586843406	117	83	34	
1592586843601	1592586843489	112	83	29	
1592586843679	1592586843574	105	83	22	
1592586843781	1592586843656	125	83	42	
1592586843845	1592586843738	107	83	24	
1592586843928	1592586843822	106	83	23	
1592586844012	1592586843906	106	83	23	
1592586844094	1592586843987	107	83	24	
1592586844178	1592586844070	108	83	25	
1592586844262	1592586844153	109	83	26	
1592586844345	1592586844236	109	83	26	
1592586844427	1592586844320	107	83	24	
1592586844515	1592586844402	113	83	30	
1592586844596	1592586844485	111	83	28	
1592586844678	1592586844569	109	83	26	
1592586844772	1592586844652	120	83	37	
1592586844835	1592586844735	100	83	17	
1592586844918	1592586844818	100	83	17	
1592586845011	1592586844901	110	83	27	
1592586845084	1592586844984	100	83	17	
1592586845167	1592586845067	100	83	17	
1592586845250	1592586845149	101	83	18	

Tabla de retraso de rotar verticalmente en Rotar

Tiempo Ubuntu	Tiempo Beagle	Diferencia	Intervalo	Diferencia	Media
1592601346834	1592601346731	103	83	20	26,2666667
1592601346926	1592601346814	112	83	29	
1592601347001	1592601346897	104	83	21	
1592601347084	1592601346980	104	83	21	
1592601347167	1592601347063	104	83	21	
1592601347250	1592601347146	104	83	21	
1592601347370	1592601347229	141	83	58	
1592601347416	1592601347312	104	83	21	
1592601347500	1592601347394	106	83	23	
1592601347583	1592601347479	104	83	21	
1592601347667	1592601347560	107	83	24	
1592601347751	1592601347648	103	83	20	
1592601347834	1592601347727	107	83	24	
1592601347917	1592601347810	107	83	24	
1592601348001	1592601347893	108	83	25	
1592601348086	1592601347976	110	83	27	
1592601348169	1592601348059	110	83	27	
1592601348252	1592601348141	111	83	28	
1592601348336	1592601348226	110	83	27	
1592601348416	1592601348308	108	83	25	
1592601348499	1592601348391	108	83	25	
1592601348582	1592601348474	108	83	25	
1592601348666	1592601348557	109	83	26	
1592601348749	1592601348640	109	83	26	
1592601348833	1592601348723	110	83	27	
1592601348919	1592601348806	113	83	30	
1592601349002	1592601348889	113	83	30	
1592601349085	1592601348971	114	83	31	
1592601349170	1592601349056	114	83	31	
1592601349251	1592601349138	113	83	30	

Tabla de retraso de rotar sobre sí mismo en Rotar

Tiempo Ubuntu	Tiempo Beagle	Diferencia	Intervalo	Diferencia	Media
1592602970867	1592602970759	108	83	25	25
1592602970950	1592602970841	109	83	26	
1592602971033	1592602970924	109	83	26	
1592602971119	1592602971008	111	83	28	
1592602971203	1592602971092	111	83	28	
1592602971283	1592602971173	110	83	27	
1592602971366	1592602971257	109	83	26	
1592602971451	1592602971340	111	83	28	
1592602971539	1592602971423	116	83	33	
1592602971618	1592602971506	112	83	29	
1592602971702	1592602971588	114	83	31	
1592602971785	1592602971671	114	83	31	
1592602971868	1592602971755	113	83	30	
1592602971950	1592602971837	113	83	30	
1592602972043	1592602971921	122	83	39	
1592602972127	1592602972003	124	83	41	
1592602972228	1592602972087	141	83	58	
1592602972284	1592602972170	114	83	31	
1592602972368	1592602972254	114	83	31	
1592602972451	1592602972336	115	83	32	
1592602972544	1592602972419	125	83	42	
1592602972619	1592602972502	117	83	34	
1592602972691	1592602972585	106	83	23	
1592602972794	1592602972667	127	83	44	
1592602972871	1592602972751	120	83	37	
1592602972950	1592602972834	116	83	33	
1592602973055	1592602972917	138	83	55	
1592602973127	1592602973000	127	83	44	
1592602973190	1592602973082	108	83	25	
1592602973273	1592602973166	107	83	24	

E.2. Mover

Tabla de retraso de mover lateralmente en Mover

Tiempo Ubuntu	Time Beagle	Diferencia	Intervalo	Diferencia	Media
1592610103149	1592610103041	108	83	25	25
1592610103239	1592610103126	113	83	30	
1592610103341	1592610103208	133	83	50	
1592610103405	1592610103290	115	83	32	
1592610103494	1592610103373	121	83	38	
1592610103559	1592610103458	101	83	18	
1592610103656	1592610103539	117	83	34	
1592610103726	1592610103623	103	83	20	
1592610103809	1592610103705	104	83	21	
1592610103892	1592610103790	102	83	19	
1592610103989	1592610103873	116	83	33	
1592610104060	1592610103955	105	83	22	
1592610104141	1592610104038	103	83	20	
1592610104232	1592610104119	113	83	30	
1592610104307	1592610104203	104	83	21	
1592610104390	1592610104286	104	83	21	
1592610104473	1592610104372	101	83	18	
1592610104559	1592610104452	107	83	24	
1592610104639	1592610104537	102	83	19	
1592610104723	1592610104618	105	83	22	
1592610104806	1592610104701	105	83	22	
1592610104889	1592610104784	105	83	22	
1592610104975	1592610104873	102	83	19	
1592610105060	1592610104950	110	83	27	
1592610105145	1592610105033	112	83	29	
1592610105220	1592610105116	104	83	21	
1592610105305	1592610105200	105	83	22	
1592610105388	1592610105282	106	83	23	
1592610105471	1592610105372	99	83	16	
1592610105559	1592610105449	110	83	27	

Tabla de retraso de mover frontalmente en Mover

Tiempo Ubnutu	Tiempo Beagle	Diferencia	Intervalo	Diferencia	Media
1592610486138	1592610486028	110	83	27	27
1592610486224	1592610486111	113	83	30	
1592610486304	1592610486194	110	83	27	
1592610486386	1592610486277	109	83	26	
1592610486472	1592610486360	112	83	29	
1592610486554	1592610486443	111	83	28	
1592610486639	1592610486527	112	83	29	
1592610486720	1592610486608	112	83	29	
1592610486803	1592610486693	110	83	27	
1592610486887	1592610486776	111	83	28	
1592610486967	1592610486858	109	83	26	
1592610487051	1592610486940	111	83	28	
1592610487142	1592610487026	116	83	33	
1592610487238	1592610487107	131	83	48	
1592610487322	1592610487190	132	83	49	
1592610487387	1592610487274	113	83	30	
1592610487471	1592610487357	114	83	31	
1592610487553	1592610487440	113	83	30	
1592610487637	1592610487522	115	83	32	
1592610487720	1592610487605	115	83	32	
1592610487803	1592610487688	115	83	32	
1592610487887	1592610487771	116	83	33	
1592610487971	1592610487854	117	83	34	
1592610488055	1592610487938	117	83	34	
1592610488137	1592610488020	117	83	34	
1592610488215	1592610488103	112	83	29	
1592610488304	1592610488185	119	83	36	
1592610488375	1592610488271	104	83	21	
1592610488468	1592610488352	116	83	33	
1592610488541	1592610488435	106	83	23	

Tabla de retraso de mover posteriormente en Mover

Timepo Ubuntu	Tiempo Bleagle	Diferencia	Intervalo	Diferencia	Media
1592613884256	1592613884163	93	83	10	10
1592613884344	1592613884245	99	83	16	
1592613884429	1592613884328	101	83	18	
1592613884511	1592613884413	98	83	15	
1592613884610	1592613884496	114	83	31	
1592613884675	1592613884578	97	83	14	
1592613884756	1592613884660	96	83	13	
1592613884850	1592613884744	106	83	23	
1592613884931	1592613884826	105	83	22	
1592613885030	1592613884909	121	83	38	
1592613885110	1592613884992	118	83	35	
1592613885173	1592613885076	97	83	14	
1592613885276	1592613885158	118	83	35	
1592613885340	1592613885241	99	83	16	
1592613885425	1592613885324	101	83	18	
1592613885506	1592613885408	98	83	15	
1592613885595	1592613885490	105	83	22	
1592613885673	1592613885573	100	83	17	
1592613885777	1592613885658	119	83	36	
1592613885840	1592613885739	101	83	18	
1592613885922	1592613885822	100	83	17	
1592613886007	1592613885905	102	83	19	
1592613886089	1592613885990	99	83	16	
1592613886176	1592613886072	104	83	21	
1592613886257	1592613886155	102	83	19	
1592613886341	1592613886238	103	83	20	
1592613886422	1592613886321	101	83	18	
1592613886507	1592613886404	103	83	20	
1592613886589	1592613886486	103	83	20	
1592613886673	1592613886569	104	83	21	

Tabla de retraso de mover hacia arriba en Mover

Tiempo Ubuntu	Tiempo Beagle	Diferencia	Intervalo	Diferencia	Media
1592615911803	1592615911711	92	83	9	9
1592615911894	1592615911794	100	83	17	
1592615911982	1592615911878	104	83	21	
1592615912078	1592615911959	119	83	36	
1592615912140	1592615912042	98	83	15	
1592615912230	1592615912126	104	83	21	
1592615912297	1592615912209	88	83	5	
1592615912380	1592615912291	89	83	6	
1592615912463	1592615912375	88	83	5	
1592615912550	1592615912458	92	83	9	
1592615912630	1592615912541	89	83	6	
1592615912735	1592615912624	111	83	28	
1592615912796	1592615912706	90	83	7	
1592615912878	1592615912790	88	83	5	
1592615912962	1592615912872	90	83	7	
1592615913045	1592615912956	89	83	6	
1592615913130	1592615913038	92	83	9	
1592615913212	1592615913122	90	83	7	
1592615913295	1592615913205	90	83	7	
1592615913395	1592615913288	107	83	24	
1592615913459	1592615913371	88	83	5	
1592615913545	1592615913454	91	83	8	
1592615913641	1592615913536	105	83	22	
1592615913712	1592615913620	92	83	9	
1592615913796	1592615913702	94	83	11	
1592615913880	1592615913786	94	83	11	
1592615913966	1592615913869	97	83	14	
1592615914046	1592615913952	94	83	11	
1592615914129	1592615914034	95	83	12	
1592615914211	1592615914119	92	83	9	

Tabla de retraso de hacia abajo en Mover

Tiempo Ubuntu	Tiempo Beagle	Diferencia	Intervalo	Diferencia	Media
1592616900464	1592616900376	88	83	5	5
1592616900552	1592616900459	93	83	10	
1592616900635	1592616900544	91	83	8	
1592616900742	1592616900625	117	83	34	
1592616900812	1592616900708	104	83	21	
1592616900884	1592616900792	92	83	9	
1592616900968	1592616900874	94	83	11	
1592616901050	1592616900958	92	83	9	
1592616901133	1592616901041	92	83	9	
1592616901216	1592616901124	92	83	9	
1592616901300	1592616901207	93	83	10	
1592616901382	1592616901292	90	83	7	
1592616901468	1592616901372	96	83	13	
1592616901548	1592616901456	92	83	9	
1592616901632	1592616901539	93	83	10	
1592616901714	1592616901621	93	83	10	
1592616901810	1592616901704	106	83	23	
1592616901881	1592616901787	94	83	11	
1592616901965	1592616901871	94	83	11	
1592616902049	1592616901954	95	83	12	
1592616902131	1592616902037	94	83	11	
1592616902215	1592616902120	95	83	12	
1592616902299	1592616902203	96	83	13	
1592616902381	1592616902286	95	83	12	
1592616902464	1592616902369	95	83	12	
1592616902548	1592616902451	97	83	14	
1592616902630	1592616902535	95	83	12	
1592616902726	1592616902617	109	83	26	
1592616902797	1592616902700	97	83	14	
1592616902881	1592616902784	97	83	14	

E.3. Libre

Tabla de retraso de desplazar frontalmente en Libre

Tiempo Ubuntu	Tiempo Beagle	Diferencia	Intervalo	Diferencia	Media
1592655161759	1592655161660	99	83	16	16
1592655161843	1592655161743	100	83	17	
1592655161926	1592655161827	99	83	16	
1592655162009	1592655161909	100	83	17	
1592655162093	1592655161994	99	83	16	
1592655162176	1592655162075	101	83	18	
1592655162261	1592655162158	103	83	20	
1592655162342	1592655162242	100	83	17	
1592655162425	1592655162325	100	83	17	
1592655162510	1592655162408	102	83	19	
1592655162593	1592655162491	102	83	19	
1592655162681	1592655162578	103	83	20	
1592655162760	1592655162657	103	83	20	
1592655162860	1592655162739	121	83	38	
1592655162926	1592655162822	104	83	21	
1592655163010	1592655162906	104	83	21	
1592655163093	1592655162989	104	83	21	
1592655163176	1592655163071	105	83	22	
1592655163260	1592655163155	105	83	22	
1592655163342	1592655163237	105	83	22	
1592655163426	1592655163322	104	83	21	
1592655163509	1592655163403	106	83	23	
1592655163593	1592655163486	107	83	24	
1592655163676	1592655163571	105	83	22	
1592655163764	1592655163654	110	83	27	
1592655163843	1592655163735	108	83	25	
1592655163952	1592655163818	134	83	51	
1592655164008	1592655163901	107	83	24	
1592655164092	1592655163986	106	83	23	
1592655164176	1592655164067	109	83	26	

Tabla de retraso de desplazar posteriormente en Libre

Tiempo Ubuntu	Tiempo Beagle	Diferencia	Intervalo	Diferencia	Media
1592658041271	1592658041172	99	83	16	16
1592658041369	1592658041258	111	83	28	
1592658041450	1592658041343	107	83	24	
1592658041543	1592658041423	120	83	37	
1592658041643	1592658041505	138	83	55	
1592658041716	1592658041588	128	83	45	
1592658041777	1592658041671	106	83	23	
1592658041870	1592658041754	116	83	33	
1592658041943	1592658041844	99	83	16	
1592658042054	1592658041920	134	83	51	
1592658042109	1592658042003	106	83	23	
1592658042192	1592658042086	106	83	23	
1592658042275	1592658042171	104	83	21	
1592658042360	1592658042252	108	83	25	
1592658042443	1592658042344	99	83	16	
1592658042538	1592658042419	119	83	36	
1592658042608	1592658042500	108	83	25	
1592658042690	1592658042586	104	83	21	
1592658042775	1592658042667	108	83	25	
1592658042857	1592658042751	106	83	23	
1592658042940	1592658042844	96	83	13	
1592658043038	1592658042917	121	83	38	
1592658043106	1592658043000	106	83	23	
1592658043189	1592658043082	107	83	24	
1592658043275	1592658043165	110	83	27	
1592658043367	1592658043249	118	83	35	
1592658043443	1592658043344	99	83	16	
1592658043546	1592658043415	131	83	48	
1592658043605	1592658043498	107	83	24	
1592658043692	1592658043580	112	83	29	

Tabla de retraso de rotar horizontalmente en Libre

Tiempo Ubuntu	Tiempo Beagle	Diferencia	Intervalo	Diferencia	Media
1592659322027	1592659321925	102	83	19	19
1592659322116	1592659322007	109	83	26	
1592659322194	1592659322090	104	83	21	
1592659322279	1592659322173	106	83	23	
1592659322361	1592659322256	105	83	22	
1592659322443	1592659322339	104	83	21	
1592659322526	1592659322423	103	83	20	
1592659322611	1592659322505	106	83	23	
1592659322693	1592659322588	105	83	22	
1592659322776	1592659322672	104	83	21	
1592659322866	1592659322755	111	83	28	
1592659322944	1592659322837	107	83	24	
1592659323046	1592659322921	125	83	42	
1592659323109	1592659323003	106	83	23	
1592659323192	1592659323087	105	83	22	
1592659323276	1592659323170	106	83	23	
1592659323359	1592659323252	107	83	24	
1592659323442	1592659323335	107	83	24	
1592659323526	1592659323418	108	83	25	
1592659323606	1592659323501	105	83	22	
1592659323694	1592659323584	110	83	27	
1592659323781	1592659323667	114	83	31	
1592659323859	1592659323751	108	83	25	
1592659323942	1592659323833	109	83	26	
1592659324026	1592659323917	109	83	26	
1592659324109	1592659323999	110	83	27	
1592659324193	1592659324082	111	83	28	
1592659324276	1592659324166	110	83	27	
1592659324359	1592659324248	111	83	28	
1592659324454	1592659324331	123	83	40	

Tabla de retraso de mover lateralmente en Libre

Tiempo Ubuntu	Tiempo Beagle	Diferencia	Intervalo	Diferencia	Media
1592664114341	1592664114232	109	83	26	26
1592664114421	1592664114315	106	83	23	
1592664114507	1592664114415	92	83	9	
1592664114605	1592664114480	125	83	42	
1592664114671	1592664114566	105	83	22	
1592664114758	1592664114647	111	83	28	
1592664114841	1592664114730	111	83	28	
1592664114925	1592664114812	113	83	30	
1592664115005	1592664114917	88	83	5	
1592664115108	1592664114979	129	83	46	
1592664115173	1592664115062	111	83	28	
1592664115256	1592664115145	111	83	28	
1592664115340	1592664115228	112	83	29	
1592664115423	1592664115311	112	83	29	
1592664115514	1592664115395	119	83	36	
1592664115589	1592664115478	111	83	28	
1592664115672	1592664115559	113	83	30	
1592664115755	1592664115644	111	83	28	
1592664115838	1592664115726	112	83	29	
1592664115918	1592664115808	110	83	27	
1592664116002	1592664115894	108	83	25	
1592664116094	1592664115976	118	83	35	
1592664116172	1592664116058	114	83	31	
1592664116255	1592664116141	114	83	31	
1592664116347	1592664116223	124	83	41	
1592664116422	1592664116307	115	83	32	
1592664116526	1592664116390	136	83	53	
1592664116588	1592664116474	114	83	31	
1592664116674	1592664116555	119	83	36	
1592664116744	1592664116639	105	83	22	

Tabla de retraso de mover hacia arriba en Libre

Tiempo Ubuntu	Tiempo Beagle	Diferencia	Intervalo	Diferencia	Media
1592665212343	1592665212241	102	83	19	19
1592665212430	1592665212323	107	83	24	
1592665212513	1592665212406	107	83	24	
1592665212595	1592665212511	84	83	1	
1592665212718	1592665212577	141	83	58	
1592665212767	1592665212656	111	83	28	
1592665212845	1592665212738	107	83	24	
1592665212927	1592665212826	101	83	18	
1592665213026	1592665212905	121	83	38	
1592665213095	1592665213002	93	83	10	
1592665213193	1592665213071	122	83	39	
1592665213260	1592665213154	106	83	23	
1592665213343	1592665213236	107	83	24	
1592665213426	1592665213319	107	83	24	
1592665213521	1592665213401	120	83	37	
1592665213593	1592665213486	107	83	24	
1592665213675	1592665213568	107	83	24	
1592665213758	1592665213651	107	83	24	
1592665213842	1592665213734	108	83	25	
1592665213926	1592665213819	107	83	24	
1592665214022	1592665213901	121	83	38	
1592665214094	1592665213984	110	83	27	
1592665214186	1592665214067	119	83	36	
1592665214259	1592665214149	110	83	27	
1592665214343	1592665214233	110	83	27	
1592665214426	1592665214315	111	83	28	
1592665214520	1592665214398	122	83	39	
1592665214593	1592665214481	112	83	29	
1592665214685	1592665214564	121	83	38	
1592665214760	1592665214647	113	83	30	

Tabla de retraso de mover hacia abajo en Libre

Tiempo Ubuntu	Tiempo Beagle	Diferencia	Intervalo	Diferencia	Media
1592669683003	1592669682892	111	83	28	36,93333
1592669683087	1592669682975	112	83	29	
1592669683171	1592669683058	113	83	30	
1592669683253	1592669683141	112	83	29	
1592669683336	1592669683224	112	83	29	
1592669683421	1592669683307	114	83	31	
1592669683511	1592669683390	121	83	38	
1592669683586	1592669683473	113	83	30	
1592669683678	1592669683556	122	83	39	
1592669683764	1592669683639	125	83	42	
1592669683857	1592669683723	134	83	51	
1592669683930	1592669683806	124	83	41	
1592669684004	1592669683887	117	83	34	
1592669684109	1592669683971	138	83	55	
1592669684170	1592669684054	116	83	33	
1592669684263	1592669684137	126	83	43	
1592669684338	1592669684220	118	83	35	
1592669684443	1592669684303	140	83	57	
1592669684505	1592669684386	119	83	36	
1592669684587	1592669684469	118	83	35	
1592669684670	1592669684553	117	83	34	
1592669684753	1592669684635	118	83	35	
1592669684837	1592669684718	119	83	36	
1592669684919	1592669684801	118	83	35	
1592669685003	1592669684884	119	83	36	
1592669685087	1592669684967	120	83	37	
1592669685170	1592669685049	121	83	38	
1592669685253	1592669685134	119	83	36	
1592669685337	1592669685216	121	83	38	
1592669685420	1592669685299	121	83	38	

Tabla de retraso de mover en diagonal en Libre

Tiempo Ubuntu	Tiempo Beagle	Diferencia	Intervalo	Diferencia	Media
1592670758647	1592670758536	111	83	28	32,666667
1592670758730	1592670758620	110	83	27	
1592670758813	1592670758703	110	83	27	
1592670758896	1592670758785	111	83	28	
1592670758979	1592670758870	109	83	26	
1592670759064	1592670758951	113	83	30	
1592670759147	1592670759035	112	83	29	
1592670759230	1592670759118	112	83	29	
1592670759311	1592670759201	110	83	27	
1592670759400	1592670759284	116	83	33	
1592670759493	1592670759367	126	83	43	
1592670759581	1592670759450	131	83	48	
1592670759646	1592670759533	113	83	30	
1592670759728	1592670759616	112	83	29	
1592670759814	1592670759699	115	83	32	
1592670759896	1592670759782	114	83	31	
1592670759982	1592670759865	117	83	34	
1592670760063	1592670759947	116	83	33	
1592670760146	1592670760032	114	83	31	
1592670760229	1592670760114	115	83	32	
1592670760313	1592670760196	117	83	34	
1592670760396	1592670760281	115	83	32	
1592670760479	1592670760362	117	83	34	
1592670760563	1592670760445	118	83	35	
1592670760655	1592670760530	125	83	42	
1592670760730	1592670760612	118	83	35	
1592670760813	1592670760694	119	83	36	
1592670760892	1592670760779	113	83	30	
1592670760980	1592670760861	119	83	36	
1592670761066	1592670760944	122	83	39	